ISSN 0542-9951

P. 1877/94

6 1994

informatyka

Zastosowania informatyki
w przemyśle oraz CAD/CAM

WYDAWNICTWO SIGMA NOT

Cena 45 000 zł

informatyka

nr 6 1994
Czerwiec
Rok wydania XXX

Miesięcznik
ISSN 0542-9951
INDEKS 36124

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
mgr inż. Piotr FUGLEWICZ
prof. dr hab. Jan GOLIŃSKI
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
mgr Krystyna
KARWICKA-RYCHLEWICZ
Władysław KLEPACZ
(redaktor naczelny)
mgr inż. Jan RYŻKO
dr Witold STANISZKIS
dr hab. Kazimierz SUBIETA
dr Zdzisław SZYJEWSKI

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek
Technicznych SIGMA NOT
Spółka z o.o.
ul. Ratuszowa 11
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych
redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu 00-950 Warszawa ul. Mazowiecka 12 telefon: 27-43-66 telefaks: 26-80-16

W numerze:

	Strona
Współpraca konkurentów – wywiad z dr. Wacławem Iszkowskim – *Krystyna Karwicka-Rychlewicz*	1
Sztuczna inteligencja w sterowaniu – *Halina Kwaśnicka, Urszula Markowska-Kaczmar*	3
Nowoczesne symulatory układów elektronicznych dla MS-Windows – *Jarosław A. Kaczyński*	10
Sieci neuronowe i logika rozmyta w systemach eksperckich – *Marcin Grzegorzewski, Dariusz Rzesiowski*	11
MegaCAD – prosty program dla wymagających – *Joanna Metelkin*	18
PVM – środowisko do przetwarzania rozproszonego w sieciach stacji roboczych – *Marek Gajęcki, Krzysztof Zieliński*	19
System CAD/CAM/CAE UNIGRAPHICS – *Andrzej Potempa*	23
BPCS – zintegrowany system zarządzania	24
Ekspercki system wspomagania sterowania produkcją – *Anna Ławrynowicz*	26

W najbliższych numerach:

Lipcowy numer INFORMATYKI będzie poświęcony dydaktyce i badaniom w zakresie informatyki ekonomicznej.

- **Edward Kolbusz i Wojciech Olejniczak omawiają problemy nauczania informatyki ekonomicznej na polskich uczelniach.**
- **Elżbieta Niedzielska zastanawia się, czy następuje zmierzch rozwoju akademickiej informatyki ekonomicznej, czy też otwiera się przed nią inna epoka.**
- **Krystyna Karwicka-Rychlewicz rozmawia o kształceniu analityków systemu z prof. Janem Golińskim, kierownikiem katedry Informatyki Gospodarczej Szkoły Głównej Handlowej w Warszawie.**
- **Piotr Sienkiewicz rozważa dylematy kształcenia informatycznego.**
- **Antoni Nowakowski omawia informatykę jako narzędzie ekonomisty.**
- **Marian Kuraś zajmuje się kształceniem analityków systemów informacyjnych.**
- **Zdzisław Gomółka omawia kształcenie ekonomistów w zakresie informatyki.**
- **Witold Abramowicz opisuje formy kształcenia w zakresie informatyki ekonomicznej w krajach niemieckojęzycznych.**
- **Zdzisław Szyjewski podaje absurdalne przykłady opodatkowania osób dokształcających się.**
- **Henryk Sroka zarysowuje wybrane problemy kształcenia w zakresie informatyki ekonomicznej.**
- **Stanisław Wrycza i Witold Rybiński piszą o teorii i praktyce nauczania systemów informatycznych.**

Warunki prenumeraty na 1994 r.

Zamówienia na prenumeratę czasopism wydawanych przez Wydawnictwo SIGMA-NOT można składać w dowolnym terminie. Mogą one obejmować dowolny okres czasu, tzn. dotyczyć dowolnej liczby kolejnych zeszytów każdego czasopisma.

Zamawiający może otrzymywać zaprenumerowany przez siebie tytuł począwszy od następnego miesiąca po dokonaniu wpłaty. Zamówienia na zeszyty sprzed daty otrzymania wpłaty będą realizowane w miarę możliwości – z posiadanych zapasów magazynowych.

Warunkiem przyjęcia i realizacji zamówienia jest otrzymanie z banku potwierdzenia dokonania wpłaty przez prenumeratora. Dokument wpłaty jest równoznaczny ze złożeniem zamówienia.

Wpłat na prenumeratę można dokonywać na ogólnie dostępnych blankietach w Urzędach Pocztowych (przekazy pieniężne) lub Bankach (polecenie przelewu), przekazując środki pod adres:
Wydawnictwo SIGMA-NOT Spółka z o.o.
Zakład Kolportażu
00-716 Warszawa, skr. poczt. 1004
konto:
PBK S.A. III 0/Warszawa nr 370015-1573-139-11

Wpłaty na prenumeratę przyjmują także wszystkie urzędy pocztowe nadawczo-odbiorcze oraz doręczyciele na terenie całego kraju

Na blankiecie wpłaty należy czytelnie podać nazwę zamawianego czasopisma, liczbę zamawianych egzemplarzy, okres prenumeraty oraz własny adres.
Na życzenie prenumeratora, zgłoszone np. telefonicznie, Zakład Kolportażu ul. Bartycka 20, 00-950 Warszawa, (telefony: 40-30-86, 40-35-89 oraz 40-00-21 wew. 249, 293, 299) wysyła specjalne blankiety zamówień wraz z aktualną listą tytułów i cennikiem czasopism.

Odbiorcy zagraniczni mogą otrzymywać czasopisma poprzez prenumeratę dewizową (wpłata dokonywana poza granicami Polski w dewizach, wg cennika dewizowego z cenami podanymi w dolarach amerykańskich) lub poprzez zamówioną w kraju prenumeratę ze zleceniem wysyłki za granicę (zamawiający podaje dokładny adres odbiorcy za granicą, dokonując równocześnie wpłaty w wysokości dwukrotnie wyższej niż cena normalnej prenumeraty krajowej).

Egzemplarze archiwalne (sprzedaż przelewowa lub za zaliczeniem pocztowym) można zamawiać pisemnie, kierując zamówienia pod adresem: Wydawnictwo SIGMA NOT, Spółka z o.o. Zakład Kolportażu, 00-716 Warszawa, ul. Bartycka 20, paw. B, tel. 40-37-31, natomiast za gotówkę można je nabyć w Klubie Prasy Technicznej w Warszawie ul. Mazowieckiej 12, tel. 26-80-17.

Istnieje możliwość zaprenumerowania 1 egz. czasopisma po cenie ulgowej przez indywidualnych członków stowarzyszeń naukowo-technicznych zrzeszonych w FSNT oraz przez uczniów zawodowych i studentów szkół wyższych. Blankiet wpłaty na prenumeratę ulgową musi być opatrzony na wszystkich odcinkach pieczęcią koła SNT lub szkoły.

W przypadku zmiany cen w okresie objętym prenumeratą Wydawnictwo zastrzega sobie prawo do wystąpienia o dopłatę różnicy cen oraz prawo do realizowania prenumeraty tylko w pełni opłaconej.
Cena jednego egzemplarza: normalna 45 000 zł, ulgowa 31 500 zł
Wartość prenumeraty w zł:
Normalna: kwartalna 135 000, półroczna 270 000, za trzy kwartały 404 000
Ulgowa: kwartalna 94 500, półroczna 189 000, za trzy kwartały 283 500

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 126/94

II okł.

P. 1877/94
POLITECHNIKA ŚLĄSKA Biblioteka Główna w Gliwicach

Współpraca konkurentów

Z dr. Wacławem Iszkowskim prezesem Polskiej Izby Informatyki i Telekomunikacji rozmawia Krystyna Karwicka-Rychlewicz

– Wśród swoich podstawowych celów PIIiT wymienia pomoc biznesową dla firm działających na naszym rynku informatyki i telekomunikacji. Ta branża dotychczas znakomicie sobie radziła bez niczyjej pomocy. Obecnie musi szukać obrońców?

– W latach 80. był to bardzo chłonny rynek, zyskowność wynosiła nawet ponad 20%. Jednocześnie firmy komputerowe, głównie handlowe, miały niewielki wpływ na to, co działo się np. w administracji, bankowości, telekomunikacji czy w dużych zakładach produkcyjnych. Obecnie ten rynek bardzo się zmienił. Są poważni odbiorcy, zmieniła się też struktura firm komputerowych. Mamy wprawdzie sporo firm małych, ale są i duże, o wielomiliardowych obrotach, utworzono przedstawicielstwa światowych firm informatycznych. To wszystko spowodowało, że rynek zaczął funkcjonować według nowych zasad. Nie osiąga się obecnie oszałamiających zysków, wpływ na sytuację wszystkich firm mają przepisy podatkowe, finansowe, celne, zasady prowadzenia przetargów, kontyngenty itp. Nawet najsilniejsza firma nie jest w stanie zadbać o zapewnienie sobie odpowiedniej ochrony finansowo-prawnej. Trzeba wziąć też pod uwagę, że w tej branży zaczęli działać ludzie, którzy – w większości przypadków – uprzednio nie mieli nic do czynienia z jakimkolwiek biznesem. Ci ludzie, przeważnie z wykształceniem technicznym nie bardzo wiedzieli, jak można dbać o własne interesy współdziałając z konkurentami, gdy w grę wchodzi np. tworzenie lobby wywierającego nacisk na prawodawców. Pierwszą próbą zebrania firm informatycznych i pokazania im, że wspólnie można osiągnąć więcej, było utworzenie Stowarzyszenia Rozwoju Systemów Otwartych. Powstanie Izby było logicznym tego następstwem.

– Czy w krajach zachodnich są odpowiedniki PIIiT?

– W każdym kraju Unii Europejskiej są tego typu organizacje; tworzą one EUROBIT. Być może z czasem i my się do niego włączymy. W St. Zjednoczonych też istnieje lobby firm komputerowych, które wywiera wpływ na rząd USA, gdy ten chce uchwalić ustawę, która by im przeszkadzała. To m.in. dzięki naciskowi firm komputerowych został np. zlikwidowany COCOM.

– Mamy już w kraju cztery organizacje grupujące firmy i ludzi związanych z informatyką. Czy nie jest to mnożenie bytów ponad potrzebę?

– W ubiegłym roku było jeszcze sporo niejasności dotyczących sfery działań tych organizacji. Wynikało to z tego, że część osób była związana z kilkoma organizacjami. Obecnie udało się te sprawy wyjaśnić i uporządkować. Osoby działające w tych organizacjach – nazwijmy je aktywem informatycznym – zostały „rozparcelowane", określony jest wyraźnie zakres ich działania. Polskie Towarzystwo Informatyczne skupia osoby z wykształceniem informatycznym, zajmuje się ich sprawami zawodowymi. Stowarzyszenie Rozwoju Systemów Otwartych, które ma duże zasługi w utworzeniu Izby, musiało nieco zreformować się. Obecnie dąży do skupienia grupy użytkowników systemów informatycznych. Chce m.in. zbierać opinię dużych użytkowników systemów informatycznych na temat tego, jak oni widzą informatyzację, omawiać wspólnie problemy, korzyści i zagrożenia. Z czasem przy Stowarzyszeniu mogłyby się afiliować istniejące już grupy użytkowników IBM, Digitala, Oracla, Novella itp. Stowarzyszenie Polski Rynek Oprogramowania PRO stawia sobie jeden główny cel – ochronę twórców oprogramowania.

– Wśród działań Izby wymienione jest prowadzenie analiz rynkowych i opracowywanie raportów gospodarczych. To bardzo ambitne zamierzenie. Jak dotąd bowiem, wiedza o naszym rynku bardziej wynika z intuicji niż z rzetelnych badań.

– Na świecie są stosowane dwie metody ilościowego badania rynku. Pierwsza – bada się użytkowników, np. pytając się, jakie jest ich zapotrzebowanie na Windows, a jakie na Novella. Tego typu badania prowadzi wiele organizacji. Druga metoda to ranking firm komputerowych, analiza podaży. Takie badania prowadzi m.in. ComputerWorld. Izba nie zakłada robienia analiz ilościowych. Dla nas większe znaczenie ma analiza jakościowa rynku. Chcemy poszukać odpowiedzi na pytanie, jaki jest stan rynku, co jest na nim pozytywne z punktu widzenia firm komputerowych, a co negatywne. Chcemy wiedzieć, jakie będzie w najbliższym czasie zapotrzebowanie na informatykę, jak te potrzeby będą definiowane, czy i na ile użytkownik będzie świadomy swoich potrzeb, jak do tego będzie przygotowany i jakiego rodzaju mogą być zewnętrzne ograniczenia dalszego rozwoju rynku informatycznego. Taką analizę jakościową Izba przygotowuje. Tego nie da się zrobić od razu, trzeba stosować metodę kolejnych przybliżeń. Wstępne przybliżenie jest już gotowe. Ostateczną wersję chcielibyśmy przedstawić na Kongresie Informatyki Polskiej, który odbędzie się pod koniec roku. Analiza jest przygotowywana nowoczesnymi metodami, nie jest to wyłącznie zbiór wypowiedzi i referatów.

– Środowisko informatyczne niepokoją trzy sprawy. Pierwsza to ochrona prawna oprogramowania, druga – ciągle aktualna – to przetargi i trzecia – kontyngenty. Jeśli chodzi o ochronę prawną oprogramowania, to uchwalenie ustawy o prawie autorskim zamyka pewien etap. Teraz trzeba ją zacząć egzekwować. (Szerzej pisaliśmy o tym w numerze majowym INFORMATYKI). O przetargach mówi się dużo i na ogół nie najlepiej. Kontrowersyjna jest też sprawa kontyngentów. Jakie jest stanowisko Izby w tych sprawach?

– Ja bym nie demonizował sprawy przetargów. O niewielkim zakupie decyduje cena i inne czynniki, np. zaufanie do firmy, pewność, że zawsze otrzyma się od niej pomoc. Jeśli chodzi o duże przetargi, prowadzone przez Bank Światowy, to stosowane są tam ich procedury i nie można zakładać, że jakieś układy i układziki mogą wpłynąć na wynik przetargu. Trzeba też wziąć pod uwagę, że przygotowanie oferty na duży przetarg wymaga zainwestowania na początek 100–200 tys. USD. Nie

każdą firmę na to stać. Ale to prawda, że trzeba wypracować jasne zasady przetargów, częściowo można oprzeć się na wzorcach zagranicznych, nie można ich jednak w pełni powielać. Przygotowywana ustawa o zamówieniach rządowych w sporej części opiera się na tym, co zostało przez nas przekazane do twórców projektu. Myślę, że ta ustawa uprości dużo spraw w sferze informatyki. Coraz więcej użytkowników jest świadomych, że kupowanie „pod stołem" grozi katastrofą. Swoją drogą, w Polsce o zakupach w ogromnym stopniu decyduje jeszcze cena. Kryterium ceny przestałoby być istotne, gdyby istniał u nas system bankowy z prawdziwego zdarzenia. Na Zachodzie nikt od razu za wszystko nie płaci. Można wziąć sprzęt i cały system na kredyt lub w leasing. Kiedy np. jest przygotowany duży system, zagraniczna firma kredytuje dostawę sprzętu, ale polska firma, robiąca aplikację, ma małe szanse wytrzymania bez zapłaty w dłuższym okresie czasu. Normalnie mogłaby iść do banku, powiedzieć, że współpracuje z dużą firmą i wówczas każdy bank powinien udzielić jej kredytu. Oczywiście nie na 50%, ale na 7–10%. Firma wtedy może taki kontrakt zrealizować, a użytkownik płaci wówczas, gdy zobaczy, że system działa.

– W ubiegłym roku po raz pierwszy wolny rynek informatyczny przestał rządzić się naturalnymi prawami – zostały wprowadzone kontyngenty.

– Izba jest przeciwnikiem kontyngentów, chociaż bierze udział w komisjach i stara się, aby kontyngenty były rozdzielane w sposób sensowny, logiczny. Pozwala to zachować pewną sprawiedliwość, jeśli w ogóle w takich przypadkach może być mowa o sprawiedliwości. Winą za wprowadzenie kontyngentów można przede wszystkim obciążać tych, którzy obłożyli komputery 20% stawką celną. W krajach Unii Europejskiej istnieje tzw. karnet Euro 1, który mówi, że jeżeli coś jest wyprodukowane w kraju należącym do Unii, to wtedy stawka celna wynosi 5%. Wszyscy powszechnie wiedzą, że PC-tów nikt tak naprawdę nie jest w stanie wytworzyć w Europie; podstawowe komponenty do komputerów pochodzą z kilku firm światowych ulokowanych w USA, bądź na Dalekim Wschodzie. Ale kilka firm europejskich dostało Euro 1, poniekąd prawem kaduka, i sprowadzało do nas sprzęt podbijając ceny. Powstał problem polskich firm, które też potrafią składać PC-ty, co nie jest rzeczą trudną. Stąd pomysł kontyngentów. W tym roku wszystko, niestety, powtórzyło się, bo nie można było wynegocjować zmiany taryfy celnej. Być może obecnie – po wejściu w życie nowych ustaleń światowych na temat stawek celnych – sprawa rozwiąże się sama. Trzeba powiedzieć, że w ub.r. kontyngenty zostały wykorzystane w ok. 80%, a więc nie były one dla kogokolwiek barierą. Natomiast sformułowanie rozporządzenia było niejasne, Izba pomogła więc MWGzZ stworzyć regulamin, który sprawę upraszczał.

– Jakie zamierzenia udało się Izbie zrealizować od czasu jej powstania (styczeń 1993 r.), a na jakie dopiero obecnie przyjdzie czas?

– Pierwszy rok był poświęcony na samoorganizowanie oraz na stworzenie pewnej struktury i możliwości funkcjonowania. Z natury rzeczy zakres działań nie był tak szeroki, jak by się chciało. Jakimś sukcesem było uporządkowanie sprawy kontyngentów. To na początku zajęło dużo czasu i energii. Udało się porozumieć z ludźmi zarządzającymi informatyką w różnych urzędach administracji państwowej oraz namówić ich, by zaczęli z nami współpracować, wysłuchiwać naszych opinii, uwzględniać je w opracowaniach prawnych.

Udało się wywrzeć pewien wpływ np. na organizację targów komputerowych, które też przeżywają kryzys. Mamy nadzieję, że każde następne targi będą lepsze, przyniosą większy pożytek jego uczestnikom. Nie da się, niestety, w najbliższym czasie zmienić tego, że w Warszawie nie ma normalnej powierzchni wystawienniczej tylko „buduary" wokół sali Kongresowej. Miasto musi wreszcie stwierdzić, że to jest potrzebne. Pozytywnym przykładem jest Kraków, gdzie zorganizowany był INFOFESTIWAL. Władze miasta doceniły korzyści płynące z goszczenia nas u siebie, dowodem tego jest chociażby udostępnienie Sukiennic. Nikt się nie bał tego, że komputery stały przy obrazach. Dzięki temu wielu młodych ludzi, którzy przyszli tam dla komputerów, obejrzało też obrazy, być może po raz pierwszy.

Nadal jesteśmy dopiero na początku układania stosunków Izby z firmami telekomunikacyjnymi. Powoli i to zaczyna się zmieniać, firmy telekomunikacyjne zaczynają dostrzegać korzyści płynące z ich współpracy na forum Izby. W tym roku spróbujemy określić, jaki jest rzeczywisty zakres ich problemów. Nikt nie wie, jakie mają być ich relacje z monopolistą, jakim jest Telekomunikacja Polska, co powinno zrobić Ministerstwo Łączności, by zmienić te relacje. Chcemy im pokazać, że można i należy współpracować z Ministerstwem Łączności przy tworzeniu nowych ustaw, że trzeba wspólnie definiować problemy. Ministerstwo też zresztą widzi, że bez reprezentacji firm, które będą stroną, nie uda się stworzyć ustawy, która byłaby w pełni, czy przynajmniej w znacznym stopniu, zaakceptowana przez wszystkich uczestników rynku telekomunikacji.

Druga podstawowa sprawa, której w ubiegłym roku nie udało się załatwić, po prostu z braku pieniędzy, to stworzenie nowej taryfy celnej. Prawdę powiedziawszy wyręczamy tutaj, w pewnym sensie, administrację państwową. Tymczasem technologia informatyczna poszła tak daleko naprzód, że obecna taryfa celna nie odpowiada rzeczywistości. GUC jest zainteresowany tym, by taka taryfa powstała, jest też zainteresowany przeszkoleniem przez nas celników, by traktowali np. ploter jako część systemu komputerowego. Istotną sprawą jest ustawa o samorządzie gospodarczym, do której też zgłaszamy swoje uwagi. Ostatnia koncepcja zakłada, że każda firma będzie musiała należeć do izby gospodarczej – branżowej lub regionalnej. Obecnie mamy 150 członków. Dla niektórych firm barierą jest składka. Szczególnie dotyczy to małych firm. Mimo wszystko jestem przekonany, że w końcu roku będzie nas więcej.

– Dziękuję za rozmowę.

Propozycja terminologiczna

Nawiązując do wypowiedzi prof. Juliusza Lecha Kulikowskiego w nr 7 INFORMATYKI z 1991 r., wielu Czytelników zwracało nam uwagę na uporczywe rozpowszechnianie w publikowanych tekstach, niezgodnego ze słowotwórczymi regułami języka polskiego, terminu *„system ekspertowy"*. Po licznych konsultacjach Redakcja postanowiła, że począwszy od obecnego numeru będzie on konsekwentnie zastępowany terminem prawidłowym, choć mało jeszcze powszechnym – *„system ekspercki"*.

HALINA KWAŚNICKA
URSZULA MARKOWSKA-KACZMAR
Centrum Informatyczne
Politechnika Wrocławska

Sztuczna inteligencja w sterowaniu

Nowoczesne metody automatyzacji procesów wymagają wspomagania komputerowego. Coraz częściej stosuje się systemy sterowania, które uznawane są za inteligentne. W systemach takich stosuje się metody pozwalające na przetwarzanie informacji nieprecyzyjnej, naśladującej zdroworozsądkowe rozumowanie (są to tzw. systemy rozmyte). Dzięki temu, sterowanie jest bardzo precyzyjne. Jeden z najbardziej złożonych systemów rozmytych to model helikoptera zbudowany przez Michio Sugeno z Instytutu Technologii w Tokio. Elementy sterujące maszyną (drążek sterowniczy, paliwo, lotki) reagują na 13 nieprecyzyjnych poleceń, takich jak: *w górę, ląduj, w miejscu*. Niektóre z tych zadań są bardzo tudne dla doświadczonego pilota, system jednak radzi sobie z nimi dobrze. Dodatkową zaletą tego systemu jest to, że polecenia mogą być wydawane głosem. Ponieważ system ten „rozumie" polecenia wydawane głosem, to potrafi na nie reagować w sposób, jak czyni to doświadczony pilot, co pozwala uznać go za „inteligentny". Ale czy system sterowania klimatyzatorem będzie inteligentny? Jakie warunki powinien spełniać system, by uznać go za inteligentny?

Encyklopedia PWN inteligencję określa następująco: ***Inteligencja*** – (psych.) *zespół zdolności umysłowych umożliwiających jednostce sprawne korzystanie z nabytej wiedzy oraz skuteczne zachowanie się wobec nowych zadań i sytuacji.*

Według tego samego źródła:

Inteligencja Sztuczna (skrót AI od angielskiego określenia *Artificial Intelligence*) – *jest to pojęcie stosowane w informatyce i oznacza rozwiązywanie problemów sposobami wzorowanymi na naturalnych działaniach i procesach poznawczych człowieka za pomocą symulujących je programów komputerowych.*

System Ekspertowy (skrót ES od ang. nazwy *Expert System*) – *komputerowy program konsultacyjny, wspomaga decyzje zastępując eksperta w danej dziedzinie, np. w diagnostyce medycznej.*

Definicji AI jest niemalże tyle, ile osób zajmujących się nią. Możliwa jest szeroka definicja [10]: *AI – dziedzina badań, które usiłują wyjaśnić i emulować zachowanie inteligentne w terminologii procesów obliczeniowych.* Zatem AI nie jest czystą nauką (część objaśniająca) ani podstawą nowatorskiej inżynierskiej dyscypliny (część emulacyjna). Wyjaśnia to interdyscyplinarną naturę sztucznej inteligencji jako dziedziny.

Podawana jest też definicja inżynierska sztucznej inteligencji, jako: *AI to generowanie reprezentacji i procedur, które automatycznie rozwiązują problemy, jakie dotychczas były rozwiązywane przez człowieka* [2, 10].

Powyższe punkty widzenia łatwo ujednolicić, jeśli przyjmiemy, że *celem AI jest zrozumienie inteligencji tak, by możliwe było jej wykorzystanie do przeprowadzania obliczeń.* Zakres problemów zawierających się w ramach AI jest dość szeroki:

Dr inż. Halina KWAŚNICKA ukończyła wydział Elektroniki Politechniki Wrocławskiej w 1976 r. W 1980 r. w Instytucie Cybernetyki Technicznej P.Wr. i uzyskała tytuł doktora nauk technicznych. Zarówno w pracy magisterskiej jak i doktorskiej zajmowała się Ewolucyjnymi Procesami Rozwoju. Od 1979 r. była zatrudniona w Ośrodku Badań Prognostycznych Politechniki Wrocławskiej, gdzie uczestniczyła w pracach dotyczących prognozowania kierunków rozwoju badań naukowych, interesując się nadal algorytmami ewolucyjnymi. Od września 1989 r. pracuje w Centrum Informatycznym Politechniki Wrocławskiej jako adiunkt naukowo-dydaktyczny. Prowadzi zajęcia dydaktyczne z podstaw informatyki i sztucznej inteligencji oraz prace dyplomowe z zastosowań algorytmów genetycznych do modelowania rynku, tworzenia uniwersalnego systemu optymalizującego i projektowania sieci neuronowych.

Dr inż. Urszula MARKOWSKA-KACZMAR ukończyła wydział Elektroniki Politechniki Wrocławskiej w 1976 r. W 1980 r. obroniła pracę doktorską w Instytucie Cybernetyki Technicznej P.Wr. z zakresu systemów operacyjnych i teorii prognozowania. Od 1980 do 1987 r. była pracownikiem Instytutu Komputerowych Systemów Automatyki i Pomiarów. W latach 1982–1983 brała udział w pracach projektowych wykonywanych przez IKSAiP i P.Wr. nad eksperymentalną siecią komputerową opartą na modelu ISO/OSI, począwszy od opracowywania koncepcji aż po oprogramowanie pierwszej wersji. Od 1987 r. jest pracownikiem dydaktycznym Centrum Informatycznego P.Wr. w zespole Podstaw Informatyki. Swoje zainteresowania naukowe skierowała ostatnio na dziedzinę sztucznej inteligencji, a szczególnie sieci neuronowych.

gry, dowody twierdzeń, rozpoznanie obrazów, przetwarzanie języków naturalnych, rozpoznawanie mowy, systemy ekspertowe (eksperckie). Przedmiotem obecnie prowadzonych prac jest wnioskowanie zdroworozsądkowe, wnioskowanie przy niepełnej i niepewnej wiedzy, projektowanie i uczenie sieci neuronowych [9].

Systemy eksperckie **(ES)** *są programami, ograniczonymi zwykle do specyficznego zakresu, próbującymi naśladować zachowanie ekspertów – ludzi.* Typowe artybuty ES:

- Wiedza jest zwykle reprezentowana w postaci deklaratywnej, umożliwiającej łatwe odczytywanie i modyfikację. Większość ES wykorzystuje struktury ***jeśli przesłanki, to akcje*** (*if-then*) – systemy oparte na regułach.
- Struktura wiedzy jest klarowna (wyłączając neuronowe ES).
- Reprezentacja wiedzy i mechanizm sterujący są rozdzielone. Często mechanizm sterujący korzysta sam z reguł (tzw. meta-reguł).
- Mają interfejsy I/O istotne z punktu widzenia użytkownika. Interfejsy te pozwalają na pracę interaktywną.
- Zawierają moduły zdobywania i modyfikacji wiedzy, które są często używane dla rozszerzenia systemów eksperckich.

Czynniki sprzyjające rozwojowi systemów eksperckich:

- Wiedza eksperta jest zasobem rzadkim i kosztownym.
- ES pozwalają większemu gronu ludzi działać w roli eksperta. Jest to użyteczne np. przy szkoleniu (służący do diagnozowania chorób bakteryjnych system MYCIN doskonale nadaje się do uczenia studentów medycyny, bez narażenia zdrowia pacjenta).
- Połączenie wiedzy kilku ekspertów może spowodować, że ES będzie działał lepiej niż pojedynczy ekspert.

Co wyróżnia systemy eksperckie w czasie rzeczywistym od konwencjonalnych ES?

Tradycyjne systemy eksperckie najczęściej znajdują zastosowania w zagadnieniach diagnostycznych i konsultacyjnych. W aplikacjach pracujących w czasie rzeczywistym, wnioskowanie najczęściej odbywa się z uwzględnieniem wielu szybko zmieniających się danych, a więc krytycznym wymogiem jest „wyprodukowanie" właściwej odpowiedzi w odpowiednim, krótkim przedziale czasu.

Coraz częściej technologie oparte na systemach eksperckich znajdują zastosowania w zadaniach czasu rzeczywistego, zwłaszcza w zadaniach szczególnie ważnych.

Systemy eksperckie pracujące w czasie rzeczywistym, w skrócie **r-t ES** (od ang. *real-time Ekspert System*) pojawiły się w odpowiedzi na potrzeby rynku. Aktualne trendy w automatyzacji produkcji, takie jak stosowanie zaawansowanych technologii (przyspieszenie procesu produkcji), zmniejszanie kosztów (zatrudnienia i szkolenia wykwalifikowanych operatorów), podwyższanie stopnia bezpieczeństwa procesu produkcyjnego itp. zainspirowały stosowanie r-t ES do sterowania procesami produkcji.

Nowoczesne urządzenia wymagają danych pochodzących z wielu sensorów, monitorowanych w małych odstępach czasu. Cyfrowe sterowniki są połączone z lokalnymi i rozległymi sieciami. Liczba danych dostępnych dla monitoringu i sterowania zwiększyła się wykładniczo. Systemy, w których są zbierane dane z 25 000–50 000 sensorów monitorujących proces nie należą do rzadkości [1].

Ilościowe dane połączone z teoretycznym modelowaniem mogą dostarczać poprawnych predykacji działania fizycznego systemu. Niestety, wymaga to zwiększenia ilości informacji koniecznej do podjęcia decyzji, zwłaszcza w nagłych sytuacjach. Informacja ta powinna być podana w postaci zrozumiałej dla operatora.

R-t ES są pisane przy współpracy najlepszych dostępnych w danej dziedzinie ekspertów. Ulepszając inteligencję systemów sterowania można znacznie zmniejszyć liczbę zatrudnionych wysokiej klasy specjalistów-operatorów, jacy są niezbędni do sterowania złożonych systemów.

Identyfikowanie aplikacji, w którym celowe i możliwe jest stosowanie r-t ES

O tym czy warto jest wdrażać r-t ES decydują dwie główne cechy:

1. Możliwość zdefiniowania zadania w sposób odpowiedni dla systemów eksperckich lub ogólnie – systemów z bazą wiedzy. Należy bowiem uwzględnić istnienie następujących faktów:

- Parametry operatorskie, które muszą być ujęte w programie monitorującym i sterującym, powinny być przystosowane do przetwarzania symbolicznego (reguły, obiekty, które są członkami klas, z dającymi się zdefiniować atrybutami). Jeśli proces jest z natury algorytmiczny, to będzie on przystosowany do obliczeń numerycznych. ES mogą czasami wymagać złożonych obliczeń numerycznych ale ich siłą jest obejmowanie rzadko występującej wiedzy (np. reguł) i jej odpowiednie stosowanie do różnych zbiorów danych.
- Ludzkie ekspertyzy są trudno dostępne.
- Istnienie w danej dziedzinie autentycznych ekspertów.
- Eksperci powinni być generalnie zgodni co do wyboru adekwatnych rozwiązań w danym problemie.
- Eksperci muszą być w stanie wyartykułować i wyjaśnić metody, jakie stosują przy rozwiązywaniu zadań z danej dziedziny. Inżynier wiedzy musi umieć „przenieść" wiedzę uzyskaną od ekspertów i dostosować ją do programu. Jeśli występuje wiele ekspertów (np. operatorów obsługujących różne maszyny), to powinni być oni generalnie zgodni w zakresie prawidłowych rozwiązań problemu. Wiele rozwiązań można zaakceptować wtedy, gdy nie są one konfliktowe i prowadzą do zgodnych rezultatów (zależnie od sytuacji niektóre z rozwiązań mogą być bardziej właściwe niż pozostałe). Istnieją różne opcje pozwalające użytkownikowi przystosować bazę wiedzy (w ustalonych ramach) do własnych upodobań tzw. *fudge factor*. Jest to szczególnie wskazane w sterowaniu procesami, w których operator wie, że sterowanie odbywa się w nienormalnych warunkach, powstałych wskutek czasowych zmian w surowcach, czynnikach środowiska (temperatura, wilgotność itp.) lub gdy należy wprowadzić krótkotrwałą zmianę w systemie sterowania, bez trwałej zmiany bazy wiedzy lub bazy reguł.
- Zadanie wymaga jedynie poznawczych (nie fizycznych) sprawności. Jeśli zadanie wymaga kombinacji sprawności poznawczej i fizycznej manipulacji, np. monitoring i sterowanie ramieniem robota na linii montażowej, to część poznawczą można uzyskać od inżyniera wiedzy, natomiast część fizyczną – metodami bardziej konwencjonalnymi.
- Trudność zadania zależy również od tego, na ile eksperci rozumieją istotę problemu. Jeśli zadanie jest tak nowe lub na tyle mało jeszcze zrozumiałe dla ekspertów, że wymaga badań podstawowych, to inżynier wiedzy niewiele zdziała.
- Budowa ES będzie niezbędna, jeśli system ma pracować w środowisku nieprzyjaznym dla człowieka, np. w zakładzie energetyki nuklearnej, stacji kosmicznej, na obcej planecie.
- Wiele problemów nadających się dla ES rozwiązuje się stosując metody heurystyczne; aby uzyskać rozwiązanie akcep-

towalne, niezbędne są metody zdroworozsądkowe. Jeśli metody konwencjonalne są zbyt trudne do realizacji, to ES mogą okazać się tam właściwe.

2. Ograniczenia aplikacji, narzucające sterowanie w czasie rzeczywistym. Mogą to być następujące sytuacje:

A. Jeśli konieczne jest, aby system działał szybciej (w sensie szybkości przetwarzania reguł) niż realizuje to konwencjonalny ES, aplikacja powinna dostarczać wiarygodnych wyników w zadanym czasie.

- Decyzja, czy powinien być stosowany r-t ES winna zależeć od stopnia, w jakim dane zmieniają się w czasie oraz od liczby konkluzji, jakie winny być osiągane po każdej zmianie wartości danej. Przykładowo, konwencjonalny ES, realizowany na średniej wielkości stacji roboczej (ang. *workstation*) może przetwarzać przeciętnie 100–1000 reguł na sekundę. Może to wystarczyć do monitorowania jednego sensora wejściowego, który jest zmieniany dziesięć razy na sekundę i powoduje „zapalanie" ok. dziesięciu reguł na każdą zmianę. R-t ES powinny przetwarzać przynajmniej 10 000 reguł na sekundę. Aby zwiększyć ogólne zdolności przetwarzania systemu, stosuje się maszyny wnioskujące połączone równolegle.

Częstym błędem przy impletowaniu r-t ES jest tworzenie prototypów reguł i funkcji koniecznych do osiągnięcia właściwych rezultatów, bez uwzględnienia szybkości zmian danych i wyników. Odpowiedź systemu może być dobra dla danych wejściowych, lecz jeśli przeskaluje się na rzeczywiste, szybko zmieniające się dane w rzeczywistym systemie oraz włączy całą bazę wiedzy, to czas odpowiedzi może okazać się nie do zaakceptowania. W tej sytuacji zaleca się staranne przygotowanie niezbędnych standardów, które testowany system powinien spełniać.

B. Jeśli wnioskowanie musi przebiegać na podstawie serii zmieniających się w czasie danych, a nie – aktualnych wartości danych.

- ES pracujące w czasie rzeczywistym wyróżniają się zdolnością analizowania danych w czasie, śledzenia trendów historycznych lub predykacji przyszłego działania na podstawie przyszłych rezultatów. Zatem baza wiedzy powinna zawierać zarówno dane aktualne, jak i historyczne.

C. Aplikacja powinna być połączona z zegarem, by system mógł „zapalać" reguły, wiązać zdarzenia lub odnosić wartości danych do aktualnego czasu (godziny). Razem z danymi w bazie wiedzy reprezentowane jest pojęcie czasu. Dostępne są informacje typu: jaka jest aktualna wartość ciśnienia w stosunku do wartości sprzed 5 minut, czy też sama wartość czasu (ustaw pozycję kurka jako otwarty o godz. 12:00:00).

D. System powinien obejmować dane asynchroniczne i zdarzenia.

E. Dane wejściowe powinny być dostarczane ze źródeł zewnętrznych, a nie przez wejścia-wyjścia I/O użytkownika.

- System powinien odczytywać dane wejściowe z sensorów zgodnie z zadaną częstotliwością właściwą dla każdej danej.

F. W systemie powinny być wbudowane takie udogodnienia, aby w widoczny sposób zwrócić uwagę na odpowiedź dotyczącą realizacji bardzo ważnych zdarzeń (np. sygnalizacja stanów alarmowych).

G. Jeśli pojawi się potrzeba, to system powinien współpracować z istniejącym oprogramowaniem (przetwarzanie sygnałów lub specyficzne aplikacje wejścia-wyjścia).

H. Aplikacja powinna być przystosowana do całodobowej pracy ciągłej, bez konserwacji systemu, utraty pamięci itp. Wymóg powyższy dotyczy wyłącznie systemów pracy ciągłej. W sterowaniu np. lotem samolotu czy kolejką podziemną możliwa jest konserwacja systemu sterowania w czasie przerwy w pracy urządzenia.

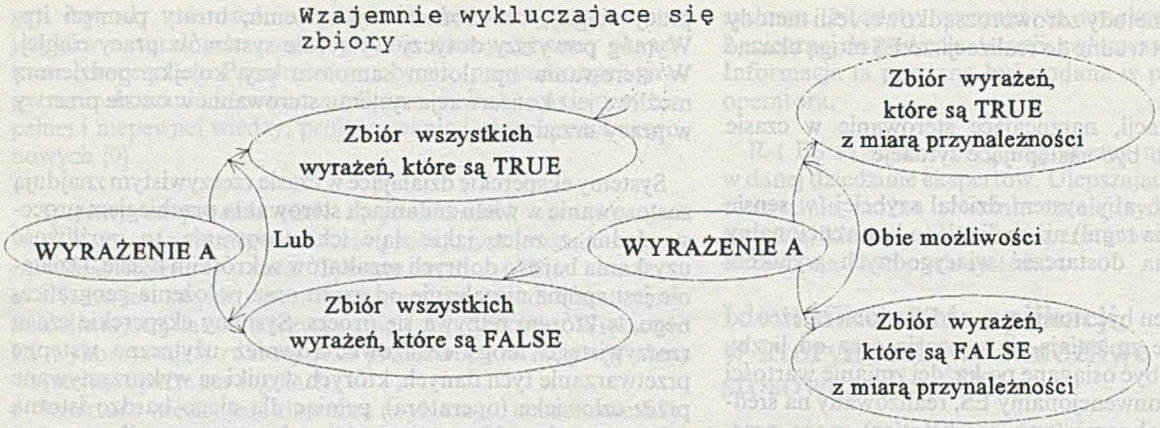
Systemy eksperckie działające w czasie rzeczywistym znajdują zastosowanie w wielu zadaniach sterowania przebiegiem procesu. Jedną z zalet, jakie daje ich stosowanie, to możliwość uzyskania bardzo dobrych rezultatów w krótkim czasie. Działanie jest spójne niezależnie od czasu oraz położenia geograficznego, w którym odbywa się proces. Systemy eksperckie czasu rzeczywistego mogą realizować również użyteczne wstępne przetwarzanie tych danych, których wyniki są wykorzystywane przez człowieka (operatora), pełniąc dla niego bardzo istotną rolę pomocniczą. Nawet tam, gdzie cała wymagana dla sterowania wiedza nie może być zredukowana do reguł czy innej, akceptowanej przez ES postaci, stosowanie systemów eksperckich może okazać się pożądane do celów odpowiedniej filtracji niezbędnej dla operatora wiedzy. Ułatwia mu to podejmowanie decyzji i może wyeliminować zmęczenie lub stan znudzenia, i tym samym przyczynić się do lepszej reakcji operatora w ważnych sytuacjach, a jednocześnie zapewnić większe bezpieczeństwo procesu. Inną korzyścią z zastosowania systemów eksperckich działających w czasie rzeczywistym do sterowania procesami jest ich zdolność do uzyskania wysokiej jakości przetwarzania często zmieniających się danych w wyniku pełniejszego monitorowania oraz inicjowania działań prewencyjnych.

Prace zmierzające do zwiększenia możliwości stosowania ES czasu rzeczywistego idą w kierunku przyspieszania działania mechanizmu wnioskującego, zarówno przez lepsze oprogramowanie, jak i zastosowanie odpowiedniej architektury.

W niektórych sytuacjach sterowanie wykorzystujące systemy eksperckie może być za mało ciągłe (zbyt „skokowe"). Obroty wentylatora można zwiększać skokowo przy zmianie temperatury otoczenia z jednego przedziału do drugiego. Próba wygładzania tych zmian przez tworzenie odpowiednio małych przedziałów wartości temperatury otoczenia jest mało opłacalna, ponieważ wprawdzie zmniejsza się skokowość sterowania, ale równocześnie wydłuża się czas obliczeń (zwiększa się liczba reguł).

Logika rozmyta

Rozumowanie komputera, zastosowane w systemach eksperckich, różni się od rozumowania przebiegającego w mózgu człowieka. „Wnioskowanie" komputera polega na przetwarzaniu ściśle zdefiniowanych faktów, które mogą być tylko prawdziwe lub fałszywe. Ludzie posługują się zdrowym rozsądkiem, który pozwala rozumieć i analizować fakty, które mogą być tylko częściowo prawdziwe. Przedział temperatury 0°C–40°C można podzielić na trzy przedziały wartości: zimno, ciepło, gorąco. Jeśli mamy temperaturę 15°C, to czy jest to jeszcze zimno czy już ciepło? Dwuwartościowa logika pozwala, by określenie „zimno" było albo prawdziwe, albo fałszywe. W rzeczywistości temperatura 15°C może oznaczać, że w pewnym stopniu jest jeszcze zimno, a w pewnym stopniu już ciepło, podobnie jak szklanka napełniona w 1/3 jest w części pełna, a w części pusta. W logice klasycznej, gdzie wartość logiczna wyrażenia może być odwzorowana w dwa wzajemnie wykluczające się zbiory (rys. 1), obiekt nie może jednocześnie należeć do zbioru i jego dopełnienia. W logice rozmytej, gdzie zbiory wyrażeń prawdziwych (*TRUE*) i fałszywych (*FALSE*) nie są rozłączne, wartość logiczna danego wyrażenia może należeć w części do zbioru zdań prawdziwych (z określoną miarą przynależności) i w części do zbioru zdań fałszywych (rys. 1b).



Rys. 1. Różnica logiki klasycznej (a) od logiki rozmytej (b)

Pierwsze badania nad logiką rozmytą i jej paradoksami zaczęły się w początkach wieku, kiedy Bertrand Russel doszedł do wniosku, że stary, grecki paradoks jest kluczem do zrozumienia teorii zbiorów i logiki matematycznej. W starej łamigłówce pewien Kreteńczyk stwierdza, że: *wszyscy Kreteńczycy kłamią*. Nasuwa się pytanie, czy wobec powyższego stwierdzenia też kłamie? Jeśli odpowiedź będzie tak, to znaczy, że mówi prawdę, a zatem kłamie. Jeśli ten Kreteńczyk mówi prawdę (odpowiedź *nie*), to znaczy, że jest kłamcą mówiąc, że wszyscy kłamią. Oba przypadki prowadzą do sprzeczności wg logiki klasycznej, ponieważ zdanie jest jednocześnie prawdziwe i fałszywe. Podobny paradoks znalazł Russel w teorii zbiorów: *zbiór wszystkich zbiorów jest zbiorem, a więc należy do samego siebie*. Z drugiej strony, zbiór wszystkich jabłek nie należy do samego siebie, bo jego elementami są jabłka, a nie zbiory. Russel postawił pytanie: czy zbiór wszystkich zbiorów, które nie należą do siebie samych, należy do siebie samego? Jeśli należy, to nie należy, jeśli zaś nie należy, to należy.

Klasyczna logika nie jest w stanie rozwiązać takich łamigłówek. Według logiki rozmytej, powyższe zdania są w 50% prawdziwe, a w 50% fałszywe.

Zbiory rozmyte lub wielowartościowe łamią prawo wykluczonego środka. Logika rozmyta nakłada tylko jedno ograniczenie: **suma przynależności danego obiektu do rozłącznych wzorów musi wynosić jeden.** Jeśli więc temperatura jest taka, że w 30% jest ciepło, to musi być w 70% nieciepło. Unika się w ten sposób sprzeczności: dany obiekt nie może być w 100% gorący i w 100% niegorący.

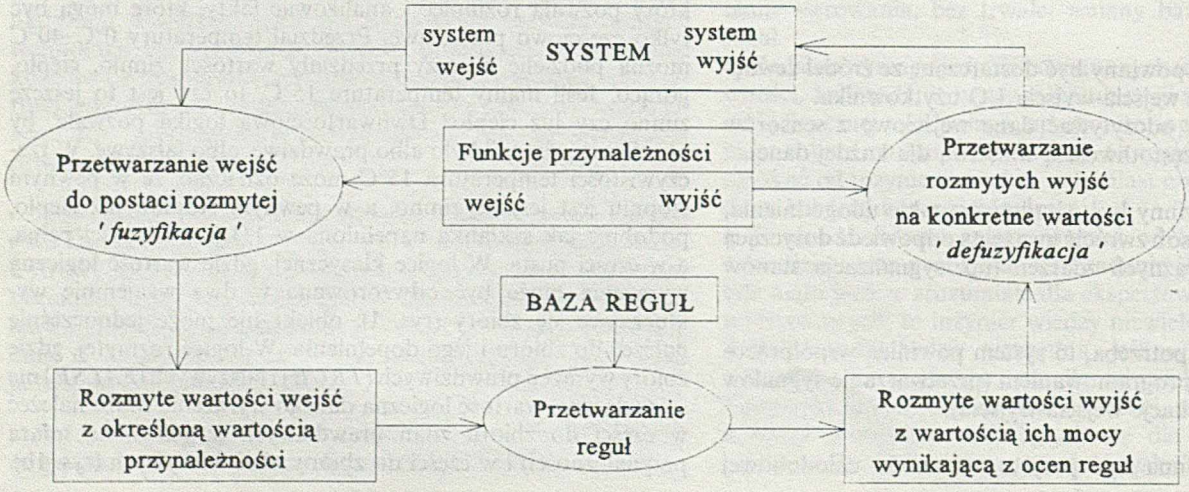
Logika rozmyta jest gałęzią informatyki, której celem jest umożliwienie komputerom tworzenia zdroworozsądkowego, zróżnicowanego obrazu niepewnej rzeczywistości. Podstawę tej dziedziny stworzyli matematycy zajmujący się logiką matematyczną, którzy w latach dwudziestych sformułowali zasadę, że wszystko można stopniować. W tym czasie polski logik, Jan Łukasiewicz, sformułował zasady logiki wielowartościowej, w której wartość zdań określa się wartościami pośrednimi między zerem a jedynką. Jeśli przez $V(p)$ oznaczymy zbiór możliwych wartości logicznych, p – liczbę wartości logicznych w pełnowartościowej logice to ograniczając wartości do przedziału $[0, 1]$ mamy:

$$V(p) = \left\{0, \frac{1}{(p-1)}, \ldots, \frac{(p-2)}{(p-1)}\, 1\right\}$$

Dla $p = 2$ będzie: $V(p) = \{0, 1\}$
Dla tak zdefiniowanego zbioru logiki wielowartościowej zdefiniowane są słabe interpretacje powiązań logicznych ($T(a)$ oznacza wartość logiczną wyrażenia a):

negacja: $T(-a) = 1 - T(a)$,
koniunkcja: $T(a \cap b) = \min\{T(a), T(b)\}$,
alternatywa: $T(a \cup b) = \max\{T(a), T(b)\}$,
implikacja: $T(a \rightarrow b) = T(\neg a \cup b)$.

Pod koniec lat trzydziestych, zajmujący się mechaniką filozof MacBlanc, zastosował logikę wielowartościową do zbiorów obiektów i wykreślił po raz pierwszy krzywe opisujące brzeg zbiorów rozmytych. Zbiory te nazwał „mętnymi".



Rys. 2. Przepływ informacji w typowym systemie sterowania z logiką rozmytą

W latach sześćdziesiątych przełomową pracę zatytułowaną „zbiory rozmyte" opublikował dziekan wydziału elektrycznego University of California, Lotfi Zadech. Nazwa „zbiory rozmyte" została powszechnie przyjęta. Zadech zastosował logikę Łukasiewicza do każdego elementu zbioru i wyprowadził pełną algebrę zbiorów rozmytych. Pierwsze próby zastosowania nastąpiły dopiero w połowie lat siedemdziesiątych: Ebrahim H. Mamdani z Queen Mary College w Londynie opracował program sterujący maszyną parową, oparty na logice rozmytej. Logika rozmyta może mieć zastosowanie nie tylko w sterowaniu, gdyż powstała raczej z potrzeb takich nauk, jak psychologia, biologia, nazywanych czasami „miękkimi naukami". Każdy proces podejmowania decyzji potencjalnie nadaje się dla logiki rozmytej.

Systemy sterowania z logiką rozmytą

Systemy wykorzystujące logikę rozmytą, to systemy ekspercke, przetwarzające reguły typu *jeśli... to...* . Reguły te przekształcają dane wejściowe – zbiór rozmyty, w dane wyjściowe – inny zbiór rozmyty. Ze zbioru wyjść rozmytych otrzymuje się ścisłe instrukcje postępowania.

Rozmyty system sterujący monitoruje, ocenia i steruje systemem zewnętrznym przez jego systemy wejść i wyjść. System wejść zawiera informacje dotyczące stanu zewnętrznego systemu, np. odczyt z czujników. Informacje wejściowe mogą wymagać wstępnego przetworzenia, np. skalowania czy normalizacji, by swoim zakresem pasowały do systemu rozmytego. Przepływ informacji w sterowaniu pokazany jest na rys. 2.

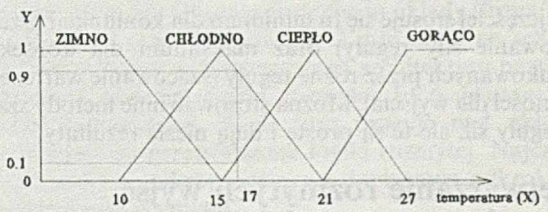
Przetwarzanie informacji wejściowych na wartości rozmyte (fuzyfikacja)

Każde wejście systemu jest skojarzone z grupą jakościowych klasyfikacji – wartości rozmytych. Przy sterowaniu klimatyzatorem daną wejściową będzie temperatura otoczenia. Informacją wejściową będzie temperatura odczytywana z miernika temperatury. Zbiorami rozmytymi dla tego wejścia mogą być: *zimno, chłodno, ciepło, gorąco*. Każda wartość mierzonej temperatury ma wartość przynależności do każdego zbioru. Może to być również zero, co będzie oznaczać, że wartość ta nie należy do danego zbioru.

Stopień przynależności do zbioru rozmytego jest definiowany przez funkcję, nazywaną zwykle funkcją przynależności (ang. *membership function*). Wyjściem w tym przykładzie mogą być obroty silnika (konkretna wartość), wyjściem rozmytym – stosowanym w regułach – mogą być zbiory rozmyte: *maksymalne, szybkie, średnie, wolne, zatrzymaj*. Proces przetwarzania konkretnej informacji wejściowej na wartość rozmytą jest to określenie wartości, z jaką przynależy ona do każdego ze zbiorów rozmytych, skojarzonych z tym wejściem.

Istotnym etapem „fuzyfikacji" wejść jest zbiór funkcji przynależności, które są definiowane przez projektanta systemu lub eksperta z danej dziedziny, na podstawie jego intuicji i doświadczenia. Funkcje przynależności bywają zmieniane kilka razy celem dostrojenia systemu tak, aby produkował pożądane wejście. Zarówno dla wejść, jak i wyjść, najczęściej stosowane są proste kszałty funkcji przynależności, trapezoidalne lub trójkątne, choć możliwe jest stosowanie również innych funkcji (np. tangens hiperboliczny). Prostszy kształt umożliwia łatwą konstrukcję tych funkcji, ich modyfikację i szybsze przetwarzanie. Funkcje te sprawdzają się w dotychczasowych zastosowaniach, co uzasadnia ich popularność.

Przykładowy kształt funkcji przynależności przedstawiony jest na rys. 3. Na rysunku tym zaznaczono punkt o temperaturze równej 17°C. Temperatura ta należy do zbioru rozmytego *ciepło* z wartością przynależności 0,1, a do zbioru *chłodno* z wartością przynależności równą 0,9. Do pozostałych zbiorów należy z wartością równą zero. O liczbie zbiorów rozmytych dla każdego wejścia decyduje ekspert. W tym przykładzie prawdopodobnie należałoby dodać jeszcze jeden zbiór nazwany *odpowiednia*, pomiędzy *chłodno* a *ciepło*, aby sterowanie było bardziej łagodne, bo zwiększyłaby się w ten sposób rozdzielczość.



Rys. 3. Przykładowa funkcja przynależności dla temperatury

Z faktu zachodzenia granic zbiorów na siebie wynika, że jedna konkretna wartość temperatury może należeć do kilku zbiorów, jednakże suma przynależności do poszczególnych zbiorów dla każdej temperatury musi być równa 1. Niejako uboczną zaletą odwzorowania konkretnych danych wejściowych na zbiory rozmyte jest to, że projektant może nazwać te zbiory używając języka naturalnego, co ułatwia opis złożonych systemów i czyni go bardziej przejrzystym.

Przetwarzanie reguł

Ekspert lub eksperci formułują reguły wnioskowania, określające wzajemne zależności między nieprecyzyjnymi określeniami wejść systemu (*zimno, ciepło*) a zarządzaniem pracą sterowanego systemu. Każda reguła składa się z części *if* ... i części *then* ... (*jeżeli ... to ...*). Część *if* składa się z jednego lub kilku warunków (przesłanek), część *then* składa się z jednej lub kilku akcji (konsekwencji). Przesłanki reguł odpowiadają wprost wartościom przynależności konkretnej danej do zbioru rozmytego, wyliczonym na etapie „fuzyfikacji" wejść. Przesłanki występujące w jednej regule są połączone funktorem logicznym **i** (*and*) albo **lub** (*or*), przy czym najczęściej jest to funktor **i**. Podobnie, akcje występujące w jednej regule są połączone najczęściej funktorem **lub**, choć mogą być również **i**.

Przykładowa reguła to:

jeżeli *cena_akcji_spada* **i** *obroty_są_duże* **to** *sprzedaj_akcje*.

Dla sterownika klimatyzatorem, przykładowe reguły to:

jeżeli jest *ciepło*, **to** obroty silnika ustaw *małe*.
jeżeli jest *chłodno*, **to** obroty silnika ustaw *średnie*.

Podczas przetwarzania reguł jest sprawdzane, czy dana reguła może być w danym momencie zastosowana, tzn. czy są spełnione jej przesłanki. Jeśli tak, to na podstawie wartości przynależności przypisanych przesłankom jest szacowana moc każdej akcji (konsekwencji) produkowanej przez rozpatrywaną regułę. Możliwe jest, że kilka reguł produkuje tę samą akcję, każda z różną siłą, należy zatem zdecydować, jaka jest ostateczna siła produkowanej akcji. Korzysta się tu z podanych wcześniej interpretacji **koniunkcji** i **alternatywy** dla logiki wielowartościowej. Ilustruje to następujący przykład:

Niech *reguła* 1: jeśli A i B to Z i X
reguła 2: jeśli C i D to Z i Y

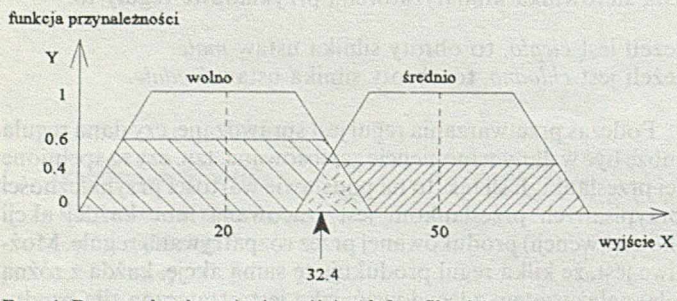
Siła_reguły_1 = min (*wartość_przynależności_A*, *wartość_przynależności_B*),
Siła_reguły_2 = min (*wartość_przynależności_C*, *wartość_przynależności_D*
Siła_akcji_X = *Siła_reguły_1*
Siła_akcji_Y = *Siła_reguły_2*
Siła_akcji_Z = max(*Siła_reguły_1*, *Siła_reguły_2*) =
= max {min (*wartość_przynależności_A*, *wartość_przynależności_B*),
min (*wartość_przynależności_C*, *wartość_przynależności_D*)}

Najczęściej stosuje się tu minimum dla koniunkcji przesłanek (szacowanie siły reguły) oraz maksimum dla konsekwencji produkowanych przez różne reguły (szacowanie wartości przynależności dla wyjścia). Można stosować inne metody szacowania reguły sił, ale te są proste i dają niezłe rezultaty.

Przetwarzanie rozmytych wyjść na konkretne wartości (defuzyfikacja)

Po zakończeniu poprzedniego etapu znana jest już siła każdej produkowanej akcji (np. w systemie prostego sterowania klimatyzatorem wyjściem są obroty silnika, a siła reguł wyznaczających wartości wyjścia może wynosić 0,4 dla *średnio* oraz 0,6 dla *szybko*). Te informacje trzeba przetworzyć na konkretną wartość obrotów silnika, np. 50 obr./min. Podobnie w przypadku rynku akcji, odpowiedź *sprzedaj_akcje* musi być ukonkretniona do liczby akcji, jakie powinny być sprzedane (niewykluczone, że pojawi się odpowiedź *sprzedaj_akcje* i w tym samym czasie z innej reguły odpowiedź *zatrzymaj_akcje*). Proces ukonkretniania wyjść pozwala znaleźć kompromisowe rozwiązanie nieprecyzyjnych i konfliktowych zdarzeń.

W systemach rozmytych można stosować różne techniki „defuzyfikacji". Techniką najczęściej stosowaną jest **metoda środka masy.** Składa się ona z kilku kroków. Dla każdego wyjścia, podobnie jak dla wejść, ekspert (lub eksperci) określają liczbę rozmytych zbiorów oraz funkcje przynależności. Metoda środka masy wykorzystuje funkcję przynależności do ukonkretniania wyjścia, na co składają się cztery następujące kroki:
1. Dla wszystkich funkcji przynależności wyjść na początku obliczane są *centroidy* (środki ciężkości pól figur płaskich).
2. Wartość funkcji przynależności wyjścia jest „obcinana" do poziomu siły, z jaką dana akcja została wyprodukowana.
3. Obliczana jest wartość pola powierzchni „obciętej" funkcji przynależności.
4. Konkretna wartość wyjścia jest obliczana jako średnia ważona wartości współrzędnej *x centroidu* i nowo obliczonego pola, z polami jako wagi.
Ilustruje to poniższy przykład oraz rys. 4.



Rys. 4. Proces ukonkretniania wejścia (defuzyfikacja)

(Na rysunku nie zachowano proporcji, a w obliczeniach przyjęto, że zbiór *wolne* obejmuje wartości *X* z przedziału 0–40, zbiór *średnie*: 30–70, w punktach obcięcia funkcji współrzędne *X* wynoszą odpowiednio: 6, 34 i 34, 66). Niech wyprodukowane akcje będą: *średnie* z siłą 0,6, i *wolne* z siłą 0,4. Pozostałe zbiory rozmyte: *maksymalne, szybkie, zatrzymaj* mają siłę 0 – nie były wyprodukowane.

Przykład

1. Centroid dla zbioru *wolne* wynosi 20, dla zbioru *średnie* jest 50.

2. Funkcja przynależności dla zbioru *wolne* jest obcięta do poziomu 0,6, dla zbioru *średnie* – do poziomu 0,4. Dla pozostałych zbiorów funkcja przynależności jest obcięta do zera.

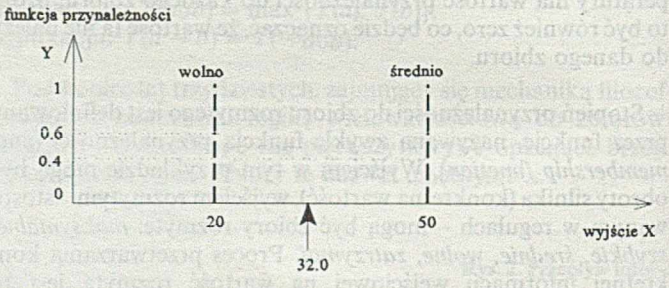
3. Pole powierzchni „obciętej" funkcji dla *wolne* wynosi: $0{,}6 \cdot (40 + 28)/2 = 20{,}4$,
Pole powierzchni „obciętej" funkcji dla *średnie* wynosi: $0{,}4 \cdot (40 + 32)/2 = 14{,}4$,

4. Wartość wyjścia $= \dfrac{20{,}4 \cdot 20 + 14{,}4 \cdot 50}{20{,}4 + 14{,}4} = 32{,}4$ obrotów/min.

Aby uprościć proces przetwarzania rozmytych wyjść na konkretne wartości stosuje się czasem zamiast trapezoidalnych funkcji przynależności – *singletony*. *Singleton* jest funkcją przynależności wyjścia reprezentowaną przez pojedyńczą linię pionową; zbiór rozmyty jest zredukowany do jednego punktu *X* na współrzędnej *x*. System taki jest w połowie systemem rozmytym, tzn. jest rozmyty od strony wejścia. Ponieważ *singleton* reprezentuje jeden punkt na osi *x*, liczenie środka masy redukuje się do liczenia średniej ważonej punktu *X* i sił reguł, z siłami reguł jako wagami (rys. 5). Dla danych z powyższego przykładu, obliczona wartość wyjścia wynosi:

Wartość wyjścia $= (0{,}6 \cdot 20 + 0{,}4 \cdot 50)/(0{,}6 + 0{,}4) = 32{,}0$.

Otrzymana wartość nie różni się dużo od poprzedniej, a cały proces ukonkretniania jest znacznie prostszy. Sytuacja taka nie jest regułą, nie zawsze dużo prostszy model z *singletonami* na wyjściach daje wyniki bardzo zbliżone do tych, jakie uzyskuje się w systemach prawdziwie rozmytych.



Rys. 5. Funkcja przynależności wyjścia w postaci singletonów

Przykłady zastosowania systemów rozmytych

Za pierwsze zastosowanie logiki rozmytej do sterowania można uznać system do sterowania piecem w cementowni, wprowadzony przez firmę z Kopenhagi F.L. Smidth & Company. Jednakże zdecydowany prym w stosowaniu systemów z logiką rozmytą w sterowaniu wiedzie Japonia. Wartość produktów zawierających sterowanie z zastosowaniem logiki rozmytej, wyprodukowanych przez japońskie firmy w 1992 r. ocenia się na kwotę 3 miliardów dolarów. Najbardziej znanym zastosowaniem i najczęściej podawanym przykładem systemu

sterowania z zastosowaniem logiki rozmytej jest sterownik wagonów metra w Sendai, w Japonii. Sterownik ten jest stosowany od końca lat osiemdziesiątych przez firmę Hitachi. Według zgodnych ocen firmy i pasażerów, sterownik ten przewyższa nie tylko standardowe sterowniki, ale i ludzką obsługę. Standardowe sterowniki uruchamiają lub zatrzymują kolejkę reagując na specjalne znaczniki, w zależności od tego jak daleko jest peron. Sterowane nimi wagony „szarpią", co nie jest przyjemne dla pasażerów, a wynika z tego, że są ściśle zaprogramowane. W ustalonej odległości od przystanku kolejka hamuje z jednakową siłą, niezależnie od tego, czy jedzie w górę, czy w dół. Pracę nad sterownikiem z systemem rozmytym rozpoczęto od zastosowania reguł logiki rozmytej do kontroli przyspieszeń i hamowania kolejki. Efektem tego była bardziej płynna jazda. Reguły uwzględniają duży zakres zmiennych, takich jak częstość i wielkości zmian prędkości, dochodzenie prędkości do największej dopuszczalnej i in. Wyniki tych prac były imponujące: obecnie stosowany system jest nie tylko bardzo dobry wg wskaźników komfortu jazdy, ale daje oszczędności w zużyciu energii o 10%. Jest stosowany nie tylko w godzinach szczytu w Sendai, ale także na niektórych liniach metra w Tokio. Właściwie japońskie metro mogłoby funkcjonować bez ludzi prowadzących składy.

Firmy zarówno koreańskie, jak i japońskie stosują systemy wykorzystujące logikę rozmytą w wielu rodzajach sprzętu. Pralki dostosowują sposób prania do masy prania, przy czym sposób prania zmienia się w miarę tego, jak zmienia się stopień zabrudzenia pranych rzeczy. W pralkach stosuje się dziesięć reguł dla dostosowania ich działania do warunków prania. Zmywarki do naczyń, które mają wbudowane takie sterowniki, reagują na stopień zmętnienia wody, odczytywany przez czujnik optyczny. Niektóre typy zmywarek mają czujnik obciążenia. W zależności od tych informacji zmieniana jest intensywność mieszania i temperatura wody.

W wideokamerach i aparatach fotograficznych systemy z logiką rozmytą są wykorzystywane do sterowania obiektywem. Canon H800, który jest na rynku od 1990 roku ma 13 reguł do ustawiania ostrości. System przetwarza dane uzyskiwane z czujników mierzących jasność obrazu z sześciu „obszarów rozmytych".

W wideokamerach Panasonic zostały zastosowane reguły, których zadaniem jest wyeliminowanie efektów drżenia ręki. To sterowanie, oparte na przewidywaniu zmian obrazu i korygowaniu go, daje lepsze efekty od sterowników opartych na ścisłych modelach matematycznych.

Odkurzacze firm Mitsubishi i Samsung dzięki zastosowaniu reguł rozmytych do sterowania ich pracą zużywają do 40% mniej energii. Parametry takiego odkurzacza, w tym jego moc ssania jest obliczana przez czterobitowy procesor na podstawie danych dotyczących przepływu kurzu. Dane te służą do określenia, na ile odkurzana powierzchnia jest już czysta.

Koncern Omron wykorzystuje systemy rozmyte do zarządzania pięcioma bazami danych systemu ochrony zdrowia. W systemie tym zastosowano 500 reguł do diagnozowania 10 tys. pacjentów. Dla każdego pacjenta system może określić indywidualne wskazania dotyczące profilaktyki zdrowotnej.

Sterowanie z logiką rozmytą stosuje się również z powodzeniem w przemyśle samochodowym. Model Saturn firmy General Motors zawiera taki sterownik do sterowania automatyczną skrzynią biegów. W pojazdach Nissana są stosowane rozmyte sterowniki w układzie zabezpieczającym przed poślizgiem przy hamowaniu, w automatycznej skrzyni biegów oraz w układzie wtrysku paliwa (opatentowane). Dawkowaniem paliwa i zapłonem steruje mikroprocesor na podstawie zestawów reguł rozmytych.

W systemach sterowania wykorzystujących logikę rozmytą stosuje się mikroprocesory przetwarzające programy oparte na algorytmach oraz czujniki mierzące parametry zewnętrzne. W 1985 r., Masaki Togai i Hiroyuki Watanabe w AT & T Bell Laboratories opracowali pierwszy mikroprocesor zaprojektowany specjalnie do zapamiętywania i przetwarzania reguł logiki rozmytej. Mikroprocesor ten przetwarzał 16 prostych reguł w czasie 12,5 mikrosekundy (0,08 mln wnioskowań logiki rozmytej na sekundę). Obecnie dostępne są układy (firma Togai InfraLogic, Inc.) przetwarzające do 2 mln reguł na sekundę. W układach tych jest wykorzystywana architektura nazywana akceleratorem obliczeń logiki rozmytej FCA (*Fuzzy Computational Acceleration*). Wiele firm pracuje nad układami dostosowanymi do przetwarzania logiki rozmytej. Najczęściej są to standardowe mikroprocesory z dopisanym mikrokodem do obróbki logiki rozmytej.

Słabym punktem systemów stosujących logikę rozmytą są reguły. Prawie wszystkie takie systemy stosują reguły podawane przez ekspertów. „Dostrajanie" zarówno reguł, jak i zbiorów rozmytych jest żmudnym etapem testowania systemu. Pożądana jest automatyzacja tego procesu. Obecnie buduje się do tego celu specjalne systemy adaptacyjne, najczęściej stosując sieci neuropodobne [6].

Systemy eksperckie z logiką rozmytą są skutecznym narzędziem do sterowania wieloma procesami rzeczywistymi. Często ich wymagania pamięciowe i obliczeniowe są mniejsze niż rozwiązywanie tego samego problemu technikami konwencjonalnymi. Prowadzone są intensywne prace nad zmniejszaniem czasu przetwarzania reguł, tworzone są specjalne mikroprocesory i specjalne systemy przyspieszające, pracujące na wejściu rozmytego systemu sterującego.

Problemem są ograniczenia sprzętowe, nie pozwalające na stosowanie uczących się sieci neuropodobnych pracujących w trybie one-line. Można oczekiwać, że jest to przyszłość tego typu systemów, gdyż próby wykorzystywania sieci do automatyzacji procesu tworzenia reguł i ich dostrajania są zachęcające.

LITERATURA

[1] Brubaker David: An Accelerated Kernel for Fuzzy Systems. AI EXPERT, March 1993, pp. 38–43

[2] Charniak Eugene, McDermott Drew: Artificial Intelligence, Addison-Wesley Publishing Comp. 1985

[3] Cox Earl: The Great Myths of Fuzzy Logic. AI EXPERT, January 1992, pp. 40–45

[4] Cox Earl: Integrating Fuzzy Logic into Neural Nets. AI EXPERT, June 1992, pp. 43–47

[5] Kosko Bart, Isaka Satoru: Logika rozmyta, tłum. Aleksy Bartnik. Świat Nauki, 9 (25), 1993, str. 60–66

[6] Kwaśnicka Halina, Markowska-Kaczmar Urszula: Wykorzystanie sieci neuropodobnych w sterowaniu (przygot. do publikacji)

[7] Levine Robert I., Drang Diane E., Edelson Barry: A Comprehensive Guide to AI and EXPERT SYSTEMS: Turbo Pascal Edition. MacGraw-Hill Book Company, 1988

[8] Lindh Lars: Fuzzy Logic Engine for Turbo Pascal. AI EXPERT, October 1993, pp. 36–41

[9] Rich Elaine, Knight Kevin: Artificial Intelligence, MacGraw-Hill Inc. 1988

[10] Schalkoff Robert J.: Artificial Intelligence: En Engineering Approach. MacGraw-Hill Inc., 1990

[11] Stein Roger: The Dempster-Shafer Theory of Evidential Reasoning. AI EXPERT, August 1993, pp. 26–30

[12] Viot Greg: Fuzzy Logic: Concepts to Constructs. AI EXPERT, November 1993, pp. 26–33

[13] Walsh Birrell: Fuzzy Logic: The American Market. AI EXPERT, November 1993, pp. 35–39

JAROSŁAW A. KACZYŃSKI

Nowoczesne symulatory układów elektronicznych dla MS-Windows

Środowisko Microsoft Windows 3.1 oferuje programistom możliwość dynamicznej wymiany danych między procesami (DDE), jednak symulatory układów elektronicznych jeszcze do niedawna nie korzystały z tego udogodnienia. Najnowsze systemy symulacji zmieniły ten stan rzeczy udostępniając swym użytkownikom szereg nowych, bardzo wygodnych funkcji.

Przypomnijmy pokrótce podstawowe grupy cyfrowych symulatorów układów elektronicznych oraz strukturę typowego systemu symulacji. Ze względu na klasę badanych układów i zastosowaną metodę symulacji możemy wyróżnić dwie zasadnicze grupy symulatorów:

- ***symulatory układów analogowych*** stosujące do budowania równań opisujących układ metodę potencjałów węzłowych lub prądów obwodowych, a do ich rozwiązywania powszechnie znane algorytmy całkowania (zwykle trapezów lub Geara). Niektóre z nich mają również możliwość symulowania układów cyfrowych w tzw. trybie mieszanym (*mixed mode simulation*). Najpopularniejsze programy z tej grupy to SPICE i NAPP.
- ***symulatory logiczne*** (zwane czasem niezbyt ściśle cyfrowymi) stosujące w opisie układów standardowy język VHDL (*VLSI Hardware Description Language*). Metody symulacji sieci połączeń zaimplementowane w tych symulatorach są zwykle opatentowane przez ich producentów. W tej grupie warto wymienić symulatory SUSIE firmy ALDEC i rodzinę symulatorów firmy ViewLogic.

Symulator - podstawowy element systemu symulacji - musi mieć dostarczony na wejście opis testowanego układu. Pierwsze symulatory zmuszały użytkowników do ręcznego tworzenia takiego opisu w postaci zbioru tekstowego. Współczesne systemy zawierają zazwyczaj graficzny ***edytor schematów*** pozwalający w znacznym stopniu zautomatyzować tę pracę. Podobnie wygląda sytuacja na wyjściu symulatora - bardziej skomplikowane operacje na wynikach testów wykonywane są zwykle w wydzielonych modułach ***postprocesorów graficznych***. Współpraca symulatora z edytorem i postprocesorem jest tą dziedziną, w której największe korzyści daje dynamiczna wymiana danych.

Ciekawym rozwiązaniem jest programowa realizacja ***sondy pomiarowej*** - narzędzia służącego do pobierania wielkości fizycznych w wybranych punktach układu i przekazywania ich do miernika lub oscyloskopu. Programowa sonda umieszczona na schemacie układu powoduje wytworzenie dynamicznego połączenia między edytorem a symulatorem, co daje możliwość obserwacji wyników bezpośrednio na schemacie.

W systemie ICAP/4Windows produkowanym przez kalifornijską firmę Intusoft i zawierającym edytor schematów SpiceNet, symulator analogowy (z możliwością pracy w trybie mieszanym) IsSpice4 oraz postprocesor graficzny IntuScope zastosowano takie rozwiązanie. Wybranie w edytorze narzędzia PROBE powoduje, że kliknięcie myszą w pewnym punkcie układu osadza na schemacie okienko z graficzną prezentacją przebiegów obliczonych w tym punkcie przez symulator.

Analogiczne rozwiązanie zastosowano w systemach symulatorów logicznych SUSIE-CAD i ACTIVE-CAD firmy Aldec z USA. Edytor graficzny Schematic Capture udostępnia w nich narzędzie PROBE umieszczające na schemacie wyniki symulacji w postaci kolorowego prostokąta zawierającego dwa symbole: kolor prostokąta oznacza aktualny stan logiczny w punkcie umieszczenia sondy, dwa symbole - ostatnie zdarzenie (zmianę stanu) w tym punkcie.

Kolejnym sposobem wykorzystania DDE jest przenoszenie do symulatora informacji o zmianach w schemacie układu.

Edytor SpiceNet z systemu ICAP/4Windows jest wyposażony w narzędzie ALTER, które powoduje wywołanie na ekran okna dialogowego umożliwiającego zmianę parametrów wskazanego elementu i uruchomienie symulacji (jeżeli na schemacie mamy już okienka sond możemy natychmiast obserwować reakcję układu na wprowadzone zmiany).

W systemach SUSIE-CAD i ACTIVE-CAD zastosowano nieco inne rozwiązanie. Proste zmiany (edycja zawartości pamięci, stanu licznika itp.) są automatycznie przenoszone do symulatora, natomiast zmiany bramek czy też połączeń między nimi powodują wygenerowanie sygnału *Power On* i doprowadzenie symulatora do stanu odpowiadającego włączeniu napięcia zasilającego układ.

Symulatory SUSIE same zapewniają wyświetlanie wyników symulacji w postaci przebiegów czasowych (w systemie nie ma oddzielnego postprocesora); wyżej opisane rozwiązania wyczerpują więc możliwości zastosowania DDE. W systemie ICAP/4Windows postprocesor graficzny IntuScope dysponuje znacznie silniejszymi narzędziami wyświetlania i obróbki wyników niż symulator IsSpice4. Dlatego też umożliwiono dynamiczne połączenie symulatora z postprocesorem i bardziej precyzyjną obserwację istotnych parametrów badanego układu.

Więcej informacji na temat omawianego oprogramowania można uzyskać u jego polskiego dystrybutora:
CADEL Sp. z o.o., ul. Tamka 38, 00-355 Warszawa,
tel. (0-22) 27-50-61 w. 252,329 fax (0-2) 635-52-62

Literatura:
1. "ICAP/4 - Design Entry & Data Analysis", Intusoft, San Pedro 1994
2. "ICAP/4 - IsSpice4", Intusoft, San Pedro 1994
3. "SUSIE™ Real-Time Interactive CAE Tools - Reference Guide", ALDEC, Newbury Park 1993

O/3/94

MARCIN GRZEGORZEWSKI
DARIUSZ RZESIOWSKI
Warszawa

Sieci neuronowe i logika rozmyta w systemach eksperckich

W ciągu ostatnich kilku lat prowadzone były intensywne badania nad zastosowaniami sieci neuronowych – neurokomputingu (ang. *neurocomputing*) i logiki rozmytej w wielu dziedzinach nauki i techniki. Doprowadziły one do stworzenia wielu aplikacji, które powszechnie udowodniły swoją skuteczność. Aplikacje te znalazły zastosowanie m.in. w automatyce i robotyce, lotnictwie, elektronice, bankowości i finansach, telekomunikacji, systemach wojskowych, medycynie, ubezpieczeniach i transporcie [4].

Bez wątpienia najsilniejszym atutem sieci neuronowych jest możliwość zastosowania ich w sytuacji, gdy zawodzą metody algorytmiczne, tzn. gdy algorytm rozwiązania zadania nie jest znany. W konsekwencji, sieci neuronowych używa się do rozwiązywania problemów, które dotychczas nie znalazły satysfakcjonujących rozwiązań.

Z kolei, układy pracujące w logice rozmytej stosuje się zazwyczaj w przypadkach, gdy ścisłe rozwiązanie zadania wymaga zbyt dużych nakładów obliczeniowych, a jednocześnie można sformułować zbiór ogólnych zasad, które prowadzą do rozwiązania w sposób przybliżony. Najbardziej popularne zastosowania tej metody to sterowniki rozmyte, które łączą prostotę konstrukcji z odpornością na zakłócenia i uszkodzenia.

W niniejszym artykule zaprezentowano niektóre sposoby wykorzystania oraz łączenia neurokomputingu i układów logiki rozmytej w systemach przetwarzania informacji.

Dla ukonkretnienia rozważań, za przykład wybrano system ekspercki. Jest to przykład, który pozwala jednak zachować cechy ogólności – zasada pracy systemu eksperckiego opartego na dostępnej bazie wiedzy oraz metody automatycznego wnioskowania są wspólne dla bardzo szerokiego zakresu zagadnień.

Jednocześnie pokazano, że wybór jednej z przedstawianych metod rozwiązywania zasad (tj. sieć neuronowa, układ logiki rozmytej lub ich połączenie) jest silnie uwarunkowany postacią dostępnej wiedzy, która może mieć formę reguł ściśle określonych, reguł nieprecyzyjnych lub zbioru danych liczbowych.

Podstawy

Preferowanym obszarem zastosowań neurokomputingu są problemy opisane zbiorem danych numerycznych, natomiast układy logiki rozmytej wymagają zestawu reguł określonych ogólnie. Umiejętne stosowanie i łączenie obu metod pozwala rozszerzyć ich obszar zastosowań i połączyć ich zalety.

Systemy eksperckie

System ekspercki korzysta z bazy wiedzy w postaci reguł (ang. *production rules*). W ogólności reguły są sformułowane w postaci implikacji:

IF (*sytuacja*) *THEN* (*akcja*)

Zdanie (*sytuacja*) to lewa strona (poprzednik) reguły, zdanie (*akcja*) to prawa strona (następnik). Ponadto system zawiera zestaw zmiennych (pamięć roboczą), który opisuje stan systemu i świata zewnętrznego. Rozwiązywanie zadań polega na zastosowaniu metody określanej jako mechanizm wnioskowania, który dokonuje porównania zawartości pamięci roboczej z lewą stroną wszystkich reguł, identyfikuje podzbiór reguł do wykonania w aktualnym stanie, wybiera jedną i wykonuje ją.
Często sposób działania mechanizmu wnioskowania jest przedstawiony w postaci trójfazowego cyklu:

Porównanie – lewa strona każdej reguły jest porównywalna z aktualnymi wartościami pamięci roboczej. W przypadku zgodności regułał jest dołączana do podzbioru reguł konkurencyjnych (ang. *conflict set*).
Wybór – następuje wybór jednej spośród podzbioru reguł konkurencyjnych.
Wykonanie – wykonanie akcji opisanej prawą stroną wybranej reguły.

Przedstawiony cykl działania opisuje sposób działania systemu eksperckiego typu *FORWARD CHAINING*. Reguły takiego systemu mają postać implikacji. Stosując klasyczny sposób wnioskowania *MODUS PONENS* i przy założeniu, że reguła – implikacja jest prawdziwa, wykonujemy akcję wtedy, gdy sytuacja jest prawdziwa – tzn. ma wartość logiczną 1.

Jeżeli możliwe jest sformułowanie reguł tak, aby zdania *sytuacja* były zdaniami logicznymi, przyjmującymi wartość 1 lub 0, to mamy do czynienia ze ścisłą reprezentacją wiedzy. Natomiast jeśli są dostępne jedynie reguły przybliżone, celowe jest wykorzystanie ukadu pracującego w logice rozmytej.

Układy logiki rozmytej

Pojęcie zbioru rozmytego jest uogólnieniem pojęcia zbioru. Zbiór rozmyty A określony na przestrzeni X jest zdefiniowany jako zbiór par uporządkowanych. Każda para składa się z elementu x i wartości funkcji przynależności $m_A(x)$. Funkcja przynależności określa, w jakim stopniu x należy do zbioru A i może przyjmować wartości z przedziału $\langle 0,1 \rangle$:

$m_A : X \to \langle 0,1 \rangle$

w odróżnieniu od funkcji przynależności określonej na zbiorze klasycznym, przyjmującej wartości 0 lub 1:

$$m_A(x) = \begin{cases} 1 \text{ gdy } x \in A \\ 0 \text{ gdy } x \notin A \end{cases}$$

Dla dwóch zbiorów rozmytych A i B, z określonymi na nich funkcjami przynależności m_A i m_B definiuje się:

- dopełnienie zbioru A jako zbiór rozmyty z funkcją przynależności:
$m'_A = 1 - m_A$
- sumę zbiorów A i B jako zbiór rozmyty z funkcją przynależności:
$m_{A \cup B} = \max(m_A, m_B)$
- iloczyn zbiorów A i B jako zbiór rozmyty z funkcją przynależności:
$m_{A \cap B} = \min(m_A, m_B)$

Sposób wnioskowania w systemach logiki rozmytej (tj. z udziałem wyrażeń logicznych, które przyjmują wyznaczone przez funkcje przynależności wartości z przedziału $\langle 0,1 \rangle$ jest określany jako *uogólniony MODUS PONENS* [5]:
dla zbiorów rozmytych A, B, $A1$ i danej reguły rozmytej:

IF (x jest A) *THEN* (y jest B)

oraz dla faktu:

x jest $A1$

można wyciągnąć wniosek:

y jest $B1$

gdzie zbiór rozmyty $B1$ jest określony następującą funkcją przynależności:

$$m_{B1}(y) = \sup_x \min[m_{A1}(x),\ I(m_A(x), m_B(y))]$$

Funkcja $I(a, b)$ jest wielowartościową implikacją stosowaną w układach logiki rozmytej. Istnieje wiele propozycji funkcji dwuargumentowych, realizujących tę funkcję np. (wg Łukasiewicza):

$$I(a, b) = \min(1, 1 - a + b)$$

która spełnia zdroworozsądkowe aksjomaty, takie jak $I(0, b) = 1$ (fałsz implikuje dowolność). Dla dalszych rozważań przyjęto popularną w aplikacjach logiki rozmytej realizację implikacji w postaci $I(a, b) = \min(a, b)$.
Uogólniony *MODUS PONENS* stanowi podstawę działania mechanizmu wnioskowania dla systemu eksperckiego działającego na bazie wiedzy rozmytej.

Reprezentacja dyskretna zbioru rozmytego A jest możliwa w postaci wektora zbudowanego z m wartości funkcji przynależności:

$$[m_A(x_1), m_A(x_2), ..., m_A(x_m)].$$

Podobnie, zbiór B można przedstawić jako wektor o p elementach:

$$[m_B(y_1), m^B(y_2), ..., m^B(y_p)].$$

Zakładamy, że oba zbiory są *normalne*, tzn. przynajmniej jedna składowa wektora reprezentującego zbiór jest równa 1. Regułę

IF (x jest A) *THEN* (y jest B),

która jest powiązaniem dwóch danych zbiorów rozmytych A i B, można przedstawić jako przekształcenie wektora A w wektor B w postaci:

$$B = A^* M$$

gdzie M jest macierzą współczynników przekształcenia o wymiarach m na p, $a'^{*'}$ przekształceniem, które jest zdefiniowane jako:

$$b_i = \max(\min(a_i, m_{ij}))$$

Dla danych A i B, czyli dla danej reguły, można wyznaczyć macierz M:

$$M = A^{T*} B$$

czyli

$$m_{ij} = \min(a^i b_j).$$

Tak zdefiniowana macierz M spełnia zależność

$$B = A^* M$$

czyli przekształca zbiór rozmyty A w zbiór rozmyty B, a dla dowolnego zbioru rozmytego $A1$, zbiór $B1$ wyznaczony zależnością $B1 = A1^* M$ spełnia uogólniony *MODUS PONENS* nadając mu postać dla zbiorów rozmytych w postaci wektorów A, B i $A1$ danej reguły rozmytej:

IF (x jest A) *Then* (y jest B)

oraz dla faktu

x jest $A1$

można wyciągnąć wniosek:

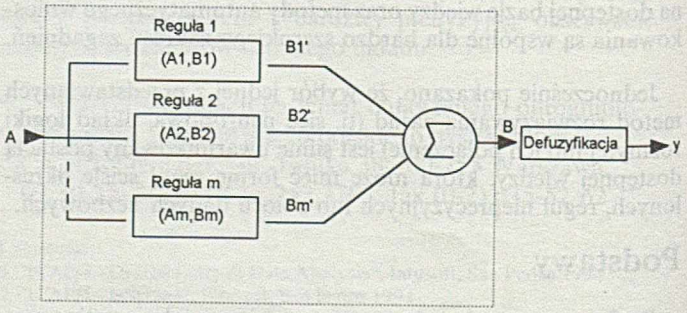
y jest $B1$

gdzie zbiór rozmyty $B1$ jest określony następująco:

$$B1 = A1^* M$$

gdzie $M = A^{T*} B$ oraz $'^{*'}$ jest przekształceniem opisanym powyżej.

Powyższe wzory stanowią podstawę koncepcji układu FAM (*Fuzzy Associative Memory*), który dokonuje przekształcenia zbiorów rozmytych w zbiory rozmyte na podstawie reguł (rys. 1).

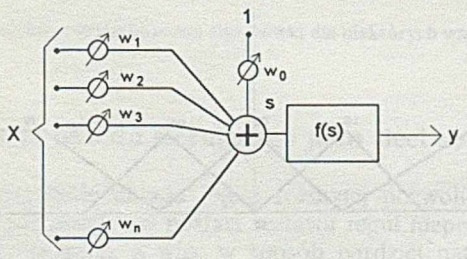


Rys. 1. Schemat działania układu FAM

Układ FAM przekształca wejściowy zbiór rozmyty A w m zbiorów $B1', ..., Bm'$ równolegle za pomocą m reguł. Zbiory $B1', ..., Bm'$ są następnie sumowane w wynikowy zbiór B, z którego można wyznaczyć, w procesie defuzyfikacji (przejście z dziedziny rozmytej w dziedzinę rzeczywistą), wartość rzeczywistą y. Układy FAM stanowią podstawę działania sterowników rozmytych, opisanych w dalszej części artykułu.

Sieci neuronowe

Sztuczna sieć neuronowa to system przetwarzania informacji, którego architektura i sposób funkcjonowania zostały zaczerpnięte z modeli biologicznych układów nerwowych [6, 10]. Sieć neuronowa składa się z dużej liczby prostych elementów przetwarzających – neuronów. Każdy neuron ma wiele wejść i jedno wyjście. Wyjście każdego neuronu jest w ogólności ważoną sumą swoich n wejść x, przekształconą przez nieliniową funkcję aktywacji $f(.)$ (rys. 2). Sygnał ten jest następnie używany jako wejście dla innych neuronów.



$$s(X) = \sum_{i=1}^{n} w_i x_i + w_0$$

$$y(X) = f(s(X))$$

Rys. 2. Budowa neuronu

Sieć neuronowa jest charakteryzowana przez typ składowych neuronów (ich funkcję aktywacji), sposób połączenia między neuronami oraz sposób, w jaki są dobierane wartości wag w_i, czyli algorytm uczenia. Celem odpowiedniej modyfikacji wartości wag jest osiągnięcie pożądanej funkcji przetwarzania sieci widzianej jako całość. Proces ten to uczenie sieci. Zwykle proces uczenia sieci rozpoczyna się z wartościami wag wybranymi losowo, przy czym podczas uczenia wagi są modyfikowane według wybranego algorytmu uczenia. Należy wyraźnie zaznaczyć, że algorytm uczenia sieci nie ma nic wspólnego z algorytmem, według którego wyuczona już sieć rozwiązuje zadany jej problem. Z reguły algorytm, którego sieć dopracowuje się podczas uczenia jest nieznany dla jej projektantów.

Podstawowe operacje wykonywane przez sieć neuronową można porównać do reakcji na bodźce. Gdy na wejście sieci wprowadzony zostaje jakiś wektor wejściowy (obraz), neurony stają się aktywne w stopniu zależnym od ich wag wejściowych i w efekcie generują odpowiedź adekwatną do wymuszenia. Taki sposób funkcjonowania typu *wymuszenie-aktywacja-odpowiedź* opisuje zarówno działanie pojedynczego neuronu, jak i całej sieci.

Sieć użyteczna, to sieć nauczona, czyli taka, która na dane wymuszenie wygeneruje dobrą odpowiedź. Procesu uczenia sieci dokonuje się przez prezentację przykładów. Uczenie może przebiegać z nauczycielem lub bez niego. Jeżeli uczenie przebiega z nauczycielem, to zestaw przykładów składa się z par: *wymuszenie-pożądana odpowiedź*. W przypadku uczenia sieci neuronowej bez nauczyciela prezentowany jest jedynie odpowiedni zestaw wymuszeń. W 80–90% praktyki inżynierskiej stosuje się uczenie z nauczycielem. W uproszczeniu procedura ta wygląda następująco:

Faza prezentacji: na wejście sieci jest podawane wymuszenie X, a sieć udziela odpowiedzi $y(X)$.
Faza porównania: odpowiedź sieci jest porównywana z odpowiedzią pożądaną $d(X)$ i jest obliczany sygnał błędu, np.: $J = (d(X) - y(X))^2$.
Faza uczenia: na podstawie sygnału błędu, w sposób zależny od wybranego algorytmu uczenia, są modyfikowane wartości wag, aby kolejna odpowiedź, stała się bliższa pożądanej.

Taki trójfazowy cykl jest stosowany dla całego zestawu przykładów i powtarzany wielokrotnie. Uczenie kończy się, gdy błąd sumaryczny dla tego zestawu zejdzie poniżej wymaganej wartości.

W przypadku uczenia bez nauczyciela pożądane odpowiedzi nie są dostępne, a sieć dokonuje samoorganizacji tak, aby osiągnąć stan stabilności energetycznej.

Po nauczeniu sieć udziela poprawnych odpowiedzi na wszystkie pytania spośród zestawu uczącego, ale nie tylko. Jej wielką zaletą jest to, że ma ona zdolność uogólniania, czyli udzielania dobrych odpowiedzi na pytania spoza zestawu uczącego, jeśli tylko zestaw uczący był reprezentatywny dla problemu danej klasy. Właśnie tu zawarta jest sztuczna inteligencja sieci – na podobne wymuszenie sieć reaguje najbardziej zbliżoną odpowiedzią, korzysta z własnych doświadczeń. Dobrym przykładem jest aproksymacja funkcji: sieć wyuczona na wybranym zestawie próbek potrafi następnie aproksymować przebieg funkcji w całej jej dziedzinie. Ogólnie, sieć neuronowa dokonuje przekształcenia R^n wymiarowej przestrzeni X, na R^m wymiarową przestrzeń Y, przy czym postać przekształcenia formowania jest w procesie uczenia.

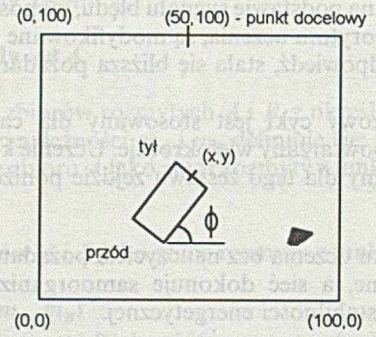
Inną zaletą sieci neuronowej jest jej krzepkość, czyli odporność na uszkodzenia. Po uszkodzeniu neuronu, część jego funkcji przejmują inne neurony sieci.

Kolejna pozytywna cecha sieci neuronowej to jej działanie równoległe – neurony pracują jako proste procesory współbieżne. Własność ta pozwala wykorzystywać sieci neuronowe do trudnych zadań obliczeniowych.

Zalety stosowania metod neurokomputingu i logiki rozmytej przy budowie konkretnych aplikacji

W momencie, gdy projektant systemu staje przed problemem wyboru sposobu rozwiązania określonego zadania, powinien on mieć na uwadze istnienie alternatywnych (w stosunku do klasycznego systemu eksperckiego) rozwiązań. Każde z nich ma swoje wady i zalety, a także preferowane zakresy zastosowań. Decydującym kryterium wyboru wydaje się być postać dostępnej wiedzy. Może być ona dostępna w postaci ściśle określonych reguł, reguł przybliżonych, czy też zestawu danych numerycznych. Zbudowanie bazy wiedzy w postaci reguł oznacza zazwyczaj konieczność skorzystania ze współpracy eksperta. Ekspert taki nie zawsze jest dostępny, ponadto istnieją sytuacje, w których ekspert posługując się intuicją i doświadczeniem nie jest w stanie ująć swej wiedzy w precyzyjne reguły. W takich przypadkach z pomocą przychodzą metody logiki rozmytej i neurokomputingu. Logika rozmyta pozwala korzystać z bazy wiedzy w postaci reguł przybliżonych. Zachowuje przy tym główną zaletę klasycznego systemu eksperckiego, tzn. zdolność wytłumaczenia odpowiedzi. Z kolei sieć neuronowa nie wymaga definiowania żadnych zasad rozwiązywania problemu, bowiem wypracowuje ona własne reguły w procesie uczenia na przykładach. Sieć neuronowa pozwala dodatkowo nawet kilkakrotnie skrócić czas budowy aplikacji. Niestety sieć neuronowa nie potrafi „uzasadnić" generowanych przez siebie odpowiedzi.

Jak efektywny może być proces budowy aplikacji z wykorzystaniem obu metod, można pokazać na klasycznym już przykładzie systemu parkującego ciężarówkę. Zadanie jest następujące [rys. 3]:



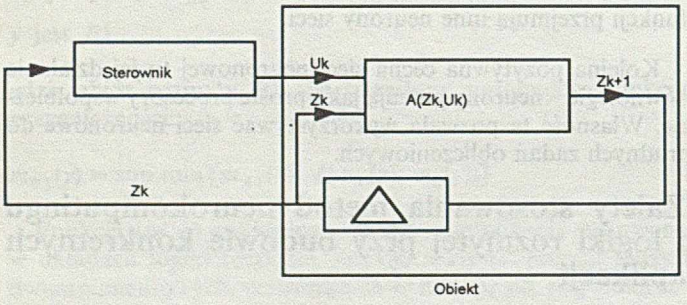
Rys. 3. Ciężarówka na placu manewrowym

• W zadanym punkcie placu manewrowego o wymiarach 100×100 znajduje się ciężarówka, ustawiona w dowolnej pozycji. Pozycję ciężarówki określają trzy zmienne:

$x: 0 \leqslant x \leqslant 100$
$y: 0 \leqslant y \leqslant 100$
$\Phi: 0 \leqslant \Phi \leqslant 360$

• Celem układu sterowania jest zaparkowanie ciężarówki tyłem w środkowym punkcie brzegu obszaru manewrowego. Zmienną sterującą jest kąt skręcenia kierownicy Θ.

$\Theta: -30 \leqslant \Theta \leqslant 30$



Rys. 4. Dyskretny w czasie układ sterowania

Schemat dyskretnego w czasie układu sterowania przedstawiono na rys. 4.

Wektor u^k reprezentuje sygnały sterujące obiektem w chwili k, wektor z_k reprezentuje w chwili k. Funkcja $A(z_k, u^k)$ wyznacza kolejny stan obiektu na podstawie jego stanu aktualnego i sygnału sterującego:

$$z(k+1) = A(z^k, u^k)$$

Zadaniem układu sterującego jest wyznaczenie właściwego sygnału sterującego u^k w każdej chwili k, tak aby osiągnąć pożądany stan końcowy z_d.

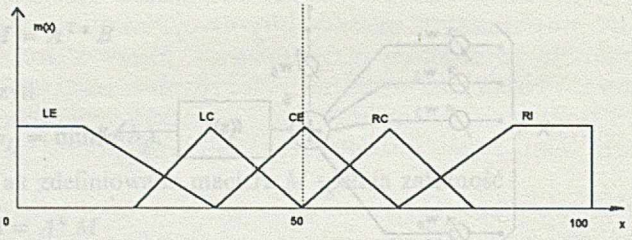
Jest to problem sterowania silnie nieliniowego. Przy użyciu metod konwencjonalnych problem ten jest trudny do rozwiązania. Typowe podejście wymagałoby odcinkowej linearyzacji modelu obiektu i następnie zbudowania sterownika na podstawie takiego modelu. Dla obiektów nieliniowych takie rozwiązanie jest złożone obliczeniowo i wymaga znacznego nakładu pracy projektowej.

Realizacja układu sterującego jako układu FAM

Układy FAM [8] stanowią podstawę działania sterowników pracujących w logice rozmytej. Ich działanie opiera się na zestawie reguł. Reguły te odwołują się do wyrażeń przybliżonych, czyli do zbiorów rozmytych.
Dla realizacji układu sterowania określono następujące wyrażenia przybliżone:

$\Phi = \{RB, RU, RV, VE, LV, LU, LB\}$
$x = \{LE, LC, CE, RC, RI\}$
$\Theta = \{NB, NM, NS, ZE, PS, PM, PB\}$

Wyrażenia te są etykietami (ang. *linguistic labels*) oznaczającymi zbiory czy też „wielkości" rozmyte, scharakteryzowane przez funkcje przynależności przyjmujące wartości z zakresu $\langle 0, 1 \rangle$. Przykład funkcji przynależności zdefiniowanych dla zbiorów rozmytych określonych dla $x \in \langle 0, 100 \rangle$ przedstawiono na rys. 5.



Rys. 5. Funkcje przynależności dla zbiorów rozmytych *LE, LC, CE, RC, RI* określonych dla $x \in \langle 0, 100 \rangle$

Reguły (oraz kształt funkcji przynależności) formułuje się na podstawie wiedzy ogólnej i doświadczenia. Zbiór reguł z użyciem wyżej zdefiniowanych wyrażeń jest przedstawiony w postaci tabelki. Reguły są w postaci:

IF (x jest *LE AND* φ jest *RB*) *THEN* (Θ jest *PS*),

tzn. każdej rozmytej parze x i φ jest przyporządkowana rozmyta wielkość sterująca Θ. Przy definicji reguł przyjęto założenie upraszczające, że wartość y nie jest istotna dla układu sterowania (zawsze jest wystarczająco dużo miejsca z tyłu samochodu do manewrowania).

$\varphi \setminus x$	LE	LC	CE	RC	RI
RB	PS	PM	PM	PB	PB
RU	NS	PS	PM	PB	PB
RV	NM	NS	PS	PM	PB
VE	NM	NM	ZE	PM	PM
LV	NB	NM	NS	PS	PM
LU	NB	NB	NM	NS	PS
LB	NB	NB	NM	NM	NS

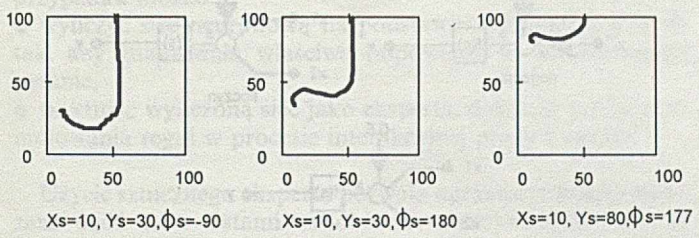
Na podstawie powyższych reguł konstruuje się układ FAM, który dla każdej pary wielkości wejściowych x i φ generuje zbiór rozmyty Θ. Rzeczywistą wartość sterowania – kąt skręcenia kierownicy Θ wyznacza się stosując „defuzyfikację" wynikowego zbioru rozmytego Θ, czyli obliczając centroid:

$$\bar{\Theta} = \frac{\sum_{j=1}^{p} \Theta_j \, m_\Theta(\Theta_j)}{\sum_{j=1}^{p} m_\Theta(\Theta_j)}$$

W celu przetestowania sprawności sterownika przeprowadzono odpowiednią symulację komputerową. Przyjęto prosty model obiektu:

$$x_{k+1} = x_k + r\cos(\varphi_{k+1})$$
$$y_{k+1} = y_k + r\sin(\varphi_{k+1})$$
$$\varphi_{k+1} = \varphi_k + \Theta_k$$

Otrzymane wyniki okazały się satysfakcjonujące: sterownik z powodzeniem parkował ciężarówkę z wielu pozycji wyjściowych przy założeniu, że było wystarczająco dużo miejsca z tyłu pojazdu (rys. 6). Popełniany błąd kąta ustawienia ciężarówki φ nie przekraczał pięciu stopni, a błąd pozycji x był mniejszy od 3.



Rys. 6. Trajektornie parkowania ciężarówki dla niektórych wartości początkowych

Realizacja układu sterującego jako sieci neuronowej

Wykorzystanie układu logiki rozmytej pozwoliło na sformalizowanie wiedzy w postaci zestawu reguł nieprecyzyjnych, nieco intuicyjnych, a więc w sposób bardziej naturalny dla człowieka. Użycie sieci neuronowej to dalszy krok na tej drodze. Informacja jaką należy dostarczyć sieci neuronowej, to reprezentatywny zestaw próbek pozbawiony jakiejkolwiek interpretacji. Dla procesu uczenia sieci nie jest istotne, czy podawany na jej wejście wektor zawiera w sobie dane o pogodzie, czy też notowania giełdowe. Cecha ta umożliwia zastosowanie sieci neuronowej do budowy systemu eksperckiego wówczas, gdy współpraca eksperta z określonych powodów (np. finansowych czy czasowych) nie jest możliwa. Częsta jest też inna sytuacja. Wielu świetnych ekspertów stojąc przed konkretnym zadaniem prawdopodobnie nie rozwiązuje go przeprowadzając ścisłe rozumowanie. Rozpoznają oni obraz, sytuację, a intuicja i doświadczenie podpowiada im właściwą odpowiedź. Może to tłumaczyć, dlaczego tak trudno jest ekspertom przedstawić zasady, jakimi się posługują w normalnej pracy. Nie jest natomiast dla nich problemem pokazać po prostu, jak znajdują odpowiedź, czyli dostarczyć próbek do procesu uczenia sieci neuronowej.

Kolejny przykład ilustruje zastosowanie sieci neuronowej w takiej właśnie sytuacji. Funkcję eksperta spełniał sterownik rozmyty z poprzedniego przykładu. Użyto go do wygenerowania zestawu próbek uczących postaci: położenie ciężarówki – kąt skręcenia kierownicy, zebranych z kilku trajektorii dojazdowych. Innymi słowy, ekspert „pokazał", jak rozwiązuje zadanie, a jego wiedza praktyczna posłużyła do nauki sieci neuronowej.

Użyto sieci, w której neurony były ułożone w dwie warstwy, 12 neuronów w pierwszej warstwie oraz jeden neuron w warstwie drugiej. Funkcją aktywacji każdego neuronu w warstwie pierwszej była funkcja sigmoidalna:

$$y = \frac{1}{1 + exp(-bs)}$$

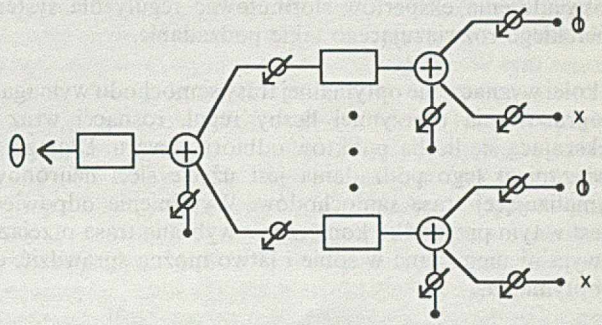
oraz funkcja liniowa – w drugiej warstwie. Jest udowodnione, że sieć o powyższej architekturze pozwala aproksymować dowolną funkcję o skończonej liczbie nieciągłości z arbitralnie wybraną dokładnością.

Zastosowany algorytm uczenia nosi nazwę algorytmu propagacji wstecznej i jest oparty na gradientowej metodzie największego spadku. Prowadzi on do minimalizacji sygnału błędu:

$$J = E(\| d(X) - y(X) \|^2),$$

gdzie $y(X)$, to odpowiedź drugiej warstwy sieci, $d(X)$ odpowiedź pożądana a E oznacza wartość oczekiwaną po wszystkich próbkach zestawu uczącego.

Przeprowadzone symulacje pokazały, iż wyuczona sieć neuronowa, po umieszczeniu jej na miejscu sterownika rozmytego, parkuje ciężarówkę nie gorzej, niż ten pierwszy (rys. 7). Opanowała ona wiedzę eksperta zawartą w próbkach uczących i wypracowała własne zasady.



Rys. 7 Sieć sterująca parkowaniem ciężarówki

Na podkreślenie zasługuje bardzo krótki czas realizacji doświadczenia. Przy użyciu programu *Neural Network Toolbox for MATLAB* firmy MathWorks Inc. przeprowadzenie całego eksperymentu zajęło kilka godzin. Jest to egzemplifikacja istotnej zalety neurokomputingu – często pozwala on znacznie skrócić czas realizacji projektu.

Na marginesie powyższych rozważań należy dodać, że znakomite rozwiązanie rzeczywistego problemu parkowania ciężarówki z przyczepą podali Nguyen i Widrow [9]. Zaproponowana przez nich metoda należy do pewnej odmiany uczenia z nauczycielem, a mianowicie uczenia stopniowanego (ang. reinforcement learning), która jest szczególnie dogodna w zagadnieniach sterowania i optymalizacji [1, 2].

Ekspert neuronowy

Ekspert neuronowy to system ekspercki łączący metody wnioskowania na podstawie reguł oraz neurokomputing. Każda z tych metod ma swoje zalety i wady. Zaletą systemu eksperckiego jest możliwość objaśnienia odpowiedzi oraz prześledzenia „toku rozumowania". Wadą jest konieczność kosztownej i uciążliwej współpracy z ekspertem w celu utworzenia bazy reguł, a także niemożność skorzystania z wiedzy nie ujętej w reguły. Zaletą sieci neuronowej jest jej praca równoległa, możliwość uczenia na podstawie próbek bez ich interpretacji, natomiast wadą niemożność uzyskania objaśnienia odpowiedzi. Ekspert neuronowy pozwala połączyć zalety obydwu metod, minimalizując ich wady.

Istnieje wiele możliwych konfiguracji. Pierwsza z nich to podział dużego zadania na mniejsze podzadania i rozwiązanie każdego najbardziej optymalną dla niego metodą. Druga, to włączenie sieci neuronowej w mechanizm działania systemu eksperckiego. Kolejną możliwością jest „połączenie równoległe" systemu eksperckiego i sieci neuronowej. Jeszcze innym rozwiązaniem może być użycie wyuczonej sieci neuronowej jako sztucznego eksperta.

Podział zadania na podzadania

Rozwiązywanie złożonego problemu wymaga często jego podziału na mniejsze podzadania oraz osobnego rozwiązania każdego z nich. Z dokonanego podziału wynika następnie, która z dostępnych metod rozwiązania jest optymalna dla danego podzadania. Takiego podejścia użyto przy budowie systemu zarządzającego rozwożeniem towaru z magazynu do odbiorców [3]. Wyróżniono tu dwa podzadania:

- załadowanie grup towarów na samochody,
- wyznaczenie optymalnej trasy dla każdego samochodu.

Rozlokowanie grup towarów na poszczególnych samochodach jest zadaniem, z którym co dzień mają do czynienia pracownicy baz przeładunkowych. Można zatem korzystając z doświadczenia ekspertów sformułować reguły dla systemu eksperckiego rozwiązującego takie podzadanie.

Z kolei wyznaczenie optymalnej trasy samochodu wymagałoby opracowania olbrzymiej liczby reguł, rosnącej wraz ze zwiększającą się liczbą punktów odbioru towaru. Dlatego też rozwiązaniem tego podzadania jest użycie sieci neuronowej optymalizującej trasę samochodów. Wyjaśnienie odpowiedzi nie jest w tym przypadku konieczne – wybrana trasa przejazdu jest wyjaśnieniem sama w sobie i łatwo można sprawdzić, czy jest optymalna.

Włączenie sieci neuronowej w mechanizm działania systemu eksperckiego

Istnieją rozwiązania, w których sieć neuronowa jest elementem funkcjonalnym systemu eksperckiego. Podczas pracy takiego systemu nieustannie zachodzi potrzeba określania, które z reguł są adekwatne do bieżącej sytuacji – stanu pamięci roboczej. Jest to zadanie porównywania obrazów (w ogólnym rozumieniu pojęcia obrazów jako zbiorów danych), którymi w tym przypadku są lewe strony reguł konkurencyjnych. W celu przyspieszenia tego procesu stosuje się także pracę wieloprocesorową.

Korzystając z równoległego charakteru przetwarzania sieci neuronowej można znacznie przyspieszyć proces wyboru reguł konkurencyjnych – sieć neuronowa potrafi dokonywać równoczesnego porównania wszystkich lewych stron reguł ze wszystkimi zmiennymi w pamięci roboczej (rys. 8).

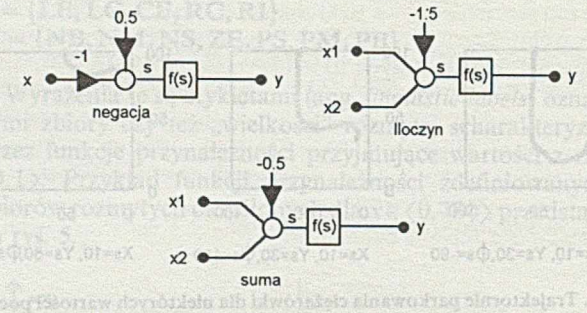
Na wejścia sieci są podawane wartości zmiennych roboczych, a wyjściem jest uporządkowany wektor binarny Y. Jedynka k-tej pozycji wektora oznacza, że k-ta reguła została zaliczona do zbioru reguł konkurencyjnych.

W pracy [11] zaproponowano zastosowanie wielowarstwowego perceptronu, w którym pojedyncze neurony realizują elementarne funkcje logiczne a w strukturze połączeń są zakodowane wyrażenia logiczne odpowiadające lewym stronom wszystkich reguł.

Funkcją aktywacji perceptronu jest funkcja progowa:

$$f(s) = \begin{cases} 1 \text{ gdy } s \geqslant 0 \\ 0 \text{ gdy } s < 0 \end{cases}$$

Sposób realizacji elementarnych funkcji logicznych negacji, sumy oraz iloczynu, odbywa się przez odpowiedni dobór wag w_i i czynnika stałego w_0 (rys. 9).



Rys. 9. Realizacja elementarnych funkcji logicznych

Tak zbudowane perceptrony stanowią cegiełki do budowy sieci realizującej złożone funkcje logiczne. W istocie więc sieć neuronowa została tu sprowadzona do roli układu kombinacyjnego, jej wagi nie są ustalane w procesie uczenia – jeżeli dostępna wiedza jest w postaci precyzyjnie określonych reguł i uczenie sieci nie jest celowe.

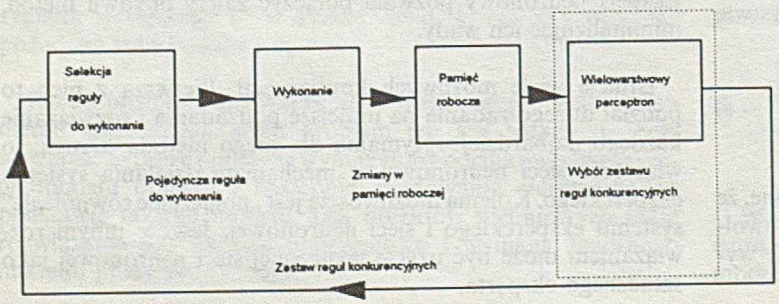
Połączenie równoległe sieci neuronowej i regulowego systemu eksperckiego

Przypadek poprzedni zakładał istnienie zestawu precyzyjnych reguł, które następnie zostały zakodowane w strukturze sieci neuronowej. Często natomiast zdarza się sytuacja następująca: dostępna wiedza obejmuje głównie ciągi danych liczbowych oraz ogólne i nieprecyzyjne reguły. Rozwiązaniem w tej sytuacji będzie równoległe połączenie sieci neuronowej i regułowego systemu eksperckiego z podziałem ról:

- sieć neuronowa udziela odpowiedzi na postawione pytania,
- wyjaśnienie odpowiedzi podaje system ekspercki na żądanie użytkownika.

Sieć pracuje tu jako klasyfikator i w odpowiedzi na obraz (zestaw danych wejściowych) zalicza go do określonej klasy oraz podaje jedną, „najlepszą" odpowiedź. Najbardziej odpowiednie do tego celu wydają się być sieci Kohonena i ART [6, 10].

Zastosowany system ekspercki pracujący w układzie *BACKWARD CHAINING* (wiązanie reguł w łańcuchach następuje w kierunku przeciwnym do kierunku wynikania logicznego) dokonuje objaśnienia (racjonalizacji) odpowiedzi sieci, dysponując jedynie ogólnie określonymi zasadami. Użyty system jest stosunkowo prosty. Sam nie jest w stanie rozwiązać całego



Rys. 8. Schemat działania systemu eksperckiego z siecią neuronową

problemu, ale może sprawdzić poprawność rozwiązania. Należy zauważyć, że naturalną implementacją takiego systemu jest układ logiki rozmytej.

Sieć neuronowa w roli eksperta

Ostatnia prezentowana konfiguracja odnosi się do sytuacji, w której cała dostępna wiedza występuje w postaci danych numerycznych. Jednocześnie istnieje potrzeba skonstruowania systemu eksperckiego pracującego na podstawie reguł, przy czym nie jest możliwa współpraca człowieka-eksperta. W takim przypadku można:

- wyuczyć sieć neuronową na podstawie dostępnych danych tak, aby znajdowała właściwą odpowiedź na każde zadane pytanie,
- traktując wyuczoną sieć jako eksperta, dokonać próby sformułowania reguł w procesie interakcyjnej pracy z siecią.

Użycie sztucznego eksperta pozwala ograniczyć koszty związane bądź to z testami na obiekcie rzeczywistym, bądź ze współpracą eksperta.

Ciekawe zastosowanie powyższej strategii można znaleźć w pracy [7]. Pokazano tam sposób w jaki sieć neuronowa może być użyta do optymalizacji sterowników rozmytych, na klasycznym przykładzie układu balansującego odwrócone wahadełko. Na podstawie odpowiednio sformułowanego modelu sterownika rozmytego została zbudowana równoważna sieć neuronowa. Po dokonaniu optymalizującego procesu uczenia sieci (ang. *temporal back propagation*) wartości odpowiednich wag wyznaczają współczynniki charakteryzujące funkcje przynależności sterownika rozmytego. W ten sposób automatycznie uzyskuje się zestaw optymalnych w danym przypadku reguł. Kluczem do podanej metody jest odwzorowanie struktury sterownika w strukturze sieci tak, aby możliwe było „wydobycie" z niej reguł. W przypadu typowej struktury sieci nie byłoby to możliwe.

* * *

Rosnące zainteresowanie układami logiki rozmytej i sieciami neuronowymi owocuje w postaci coraz to nowych zastosowań. Metody te są szczególnie dogodne wszędzie tam, gdzie dostępna wiedza ma postać reguł nieprecyzyjnych, bądź jedynie zestawu danych numerycznych. Bardzo obiecujące wydaje się być połączenie obydwu metod w ramach jednego systemu przetwarzania informacji, gdyż każda z nich podejmuje problem projektowania systemów „inteligentnych" w odmienny sposób. Sieci neuronowe zasadniczo stanowią algorytm obliczeniowy niskiego poziomu oferujący wysoką efektywność prac z nieprzetworzonymi danymi pomiarowymi. Z drugiej strony logika rozmyta to sposób reprezentacji, przetwarzania i wykorzystania informacji zawierających niepewność o charakterze niestatycznym [8]. Tak więc logika rozmyta operuje na wyższym (semantycznym czy lingwistycznym) poziomie niż sieci neuronowe. W konsekwencji obie techniki są często dla siebie komplementarne. Sieci neuronowe dostarczając zdolność przetwarzania dużej liczby danych pomiarowych i logika rozmyta nadając ramy strukturalne dla informacji częściowo przetworzonych. Istnieje wiele możliwych konfiguracji, w których jedna z omawianych technik może być wkomponowana w system zrealizowany przy użyciu drugiej. Ilustracja tego faktu na przykładzie systemu eksperckiego została przedstawiona na niniejszym artykule.

Zarówno sieci neuronowe, jak i układy logiki rozmytej są dynamicznymi systemami adaptacyjnymi, a proces ich adaptacji jest kształtowany czy też „programowany" przez zestawy danych numerycznych lub lingwistycznych. Obie metody, w przeciwieństwie do estymatorów statystycznych, przybliżają pożądaną funkcję przetwarzania bez matematycznego modelu określającego zależność wyjścia od wejścia układu. W tym sensie są one estymatorami bezmodelowymi (ang. *model-free estimators*), uczą się na podstawie doświadczeń. Ma to istotny wpływ na proces budowy aplikacji, który przebiega zasadniczo odmiennie od realizacji klasycznego, algorytmicznego systemu informatycznego. Następuje przesunięcie nakładów projektowych na inne etapy realizacji oraz znaczna ich redukcja, a w przypadku neurokomputingu, całkowite wyeliminowanie fazy kodowania, która stanowi jedno z głównych źródeł błędów oprogramowania. Zobrazowaniem tego jest zaprezentowana procedura konstrukcji sterownika parkującego ciężarówkę.

LITERATURA

[1] Barto A.G., Sutton R.S. and Anderson C.W.: Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Trans. Syst. Man. Cybern. Vol. SMC-13, No. 5, 1983 pp. 834–846

[2] Berenji H.R., Khedkar P.: Learning and Tuning Fuzzy Logic Controllers Through Reinforcements. IEEE Trans. on Neural Networks. Vol. 3. No. 5, September 1992

[3] Caudill M.: Expert Networks. BYTE, October 1991

[4] DARPA Neural Study Lexington, MA: Mit Lincoln Laboratory, 1988

[5] Dubois D., Prade H.: Fuzzy Sets and Systems: Theory and Applications. Academic Press, Orlando, FL, 1980

[6] Hecht-Nielsen R.: Neurocomputing. Addison-Wesley Publishing, Reading MA 1990

[7] Jyh-Shing R. Jang: Self-Learning Fuzzy Controllers Based on Temporal Back Propagation. IEEE Trans. on Neural Networks. Vol. 3, No 5, September 1992

[8] Kosko B.: Neural Networks and Fuzzy Systems. Prentice-Hall International, London, 1992

[9] Nguyen D., Widrow B.: Neural Networks for Self-Learning Control Systems. IEE Control Systems Magazine, April 1990

[10] Tadeusiewicz R.: Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa, 1993

[11] Sartori M.A., Passino K.M., Antsaklis P.J.: A Multilayer Perceptron to the Match Phase Problem in Rule-Based Artificial Intelligence Systems. IEEE Trans. on Knowledge and Data Engineering. Vol. 4, No. 3, June 1992.

Ogłoszenia prosimy zgłaszać:

pisemnie
Red. INFORMATYKI
Pl. Inwalidów 10, p. 104, 01-552 Warszawa
lub
Dział Reklamy i Marketingu
ul. Mazowiecka 12, 00-950 Warszawa
skr. poczt. 1004

telefonicznie
nr 39-14-34 lub 27-43-66
telefaksem
nr 26-80-16
teleksem
814550
z zaznaczeniem – Red. INFORMATYKI

MegaCAD – prosty program dla wymagających

MegaCAD to profesjonalny system wspomagający tworzenie rysunków technicznych.

Na terenie Niemiec pracuje już ponad 20 tys. jego komercyjnych wersji, m.in. w tak znanych firmach jak BMW, Siemens, Audi. Również w Polsce szybko wzrasta jego popularność.

Sukces programu jest wynikiem jego licznych zalet:

1. Polskojęzyczność – komunikaty, opisy, podpowiedzi, helpy zostały przetłumaczone na język polski;

2. Uniwersalność sprzętowa – system został zaprojektowany z myślą o wykorzystywaniu go do pracy na dowolnym komputerze od XT wzwyż;

3. Łatwość obsługi – największy nacisk przy opracowywaniu programu firma MegaTECH położyła na stworzenie systemu możliwie najbardziej przyjaznego dla użytkownika. Całość obsługi oparto na ikonach, dzięki czemu wszystkie polecenia są czytelne i łatwe do zapamiętania;

4. Łatwość nauki – już po dziesięciu godzinach nauki od pierwszego (w życiu) uruchomienia komputera, użytkownik jest w stanie wykonywać pełnowartościowe projekty;

5. Szybkość działania – wyniki testów przeprowadzonych przez czasopismo DOS TEST uplasowały MegaCAD-a na pierwszym miejscu wśród systemów CAD;

6. Komunikacja – system pozwala na wymianę rysunków z dowolnym programem CAD przez formaty: DXF, HPGL, CDL, IGES, ASCII;

7. Bogactwo funkcji rysujących – najlepszym tego przykładem jest 25 różnych możliwości rysowania linii;

8. Możliwości edycji – olbrzymia paleta opcji, począwszy od kopiowania, obracania, skalowania, wycinania, a skończywszy na wszelkiego rodzaju zmianach w istniejących rysunkach;

9. Zestawienia – udostępniono użytkownikowi prostą komunikację z bazami danych oraz możliwość automatycznego generowania bezpośrednio na rysunku wykazów użytych elementów;

10. Plot – program umożliwiający uzyskanie rysunku na dowolnej drukarce (lub ploterze) w żądanej rozdzielczości i wybranym formacie;

11. Różnorodność wersji – od podstawowej wersji płaskiej do pełnej obiektowej – trójwymiarowej, od systemu DOS przez WINDOWS, WINDOWS NT do sieci NOVELL;

12. Kieszeniodostępność – ceny już od 7.500.000 zł;

13. Uniwersalność zastosowań – do każdego działu techniki istnieją bogate biblioteki typowych elementów, niektóre z nich powstały już z uwzględnieniem PN i wymagań polskich biur projektowych;

14. Oszczędność miejsca na dysku – rysunki zajmują kilkakrotnie mniej miejsca w pamięci niż wykonane innym programem;

15. Perspektywy rozwoju – firma MegaTECH systematycznie udoskonala podstawowy pakiet oprogramowania, a polski dystrybutor stale opracowuje nowe nakładki branżowe i dodatkowe procedury;

16. Edukacja – informujemy, że polskie placówki oświatowe mogą otrzymać pełną wersję systemu, pokrywając jedynie koszty transportowe i materiałowe (tj. 500.000 zł);

17. Popularyzacja – potencjalny użytkownik programu może poznać jego zalety posługując się wersjami treningowymi.

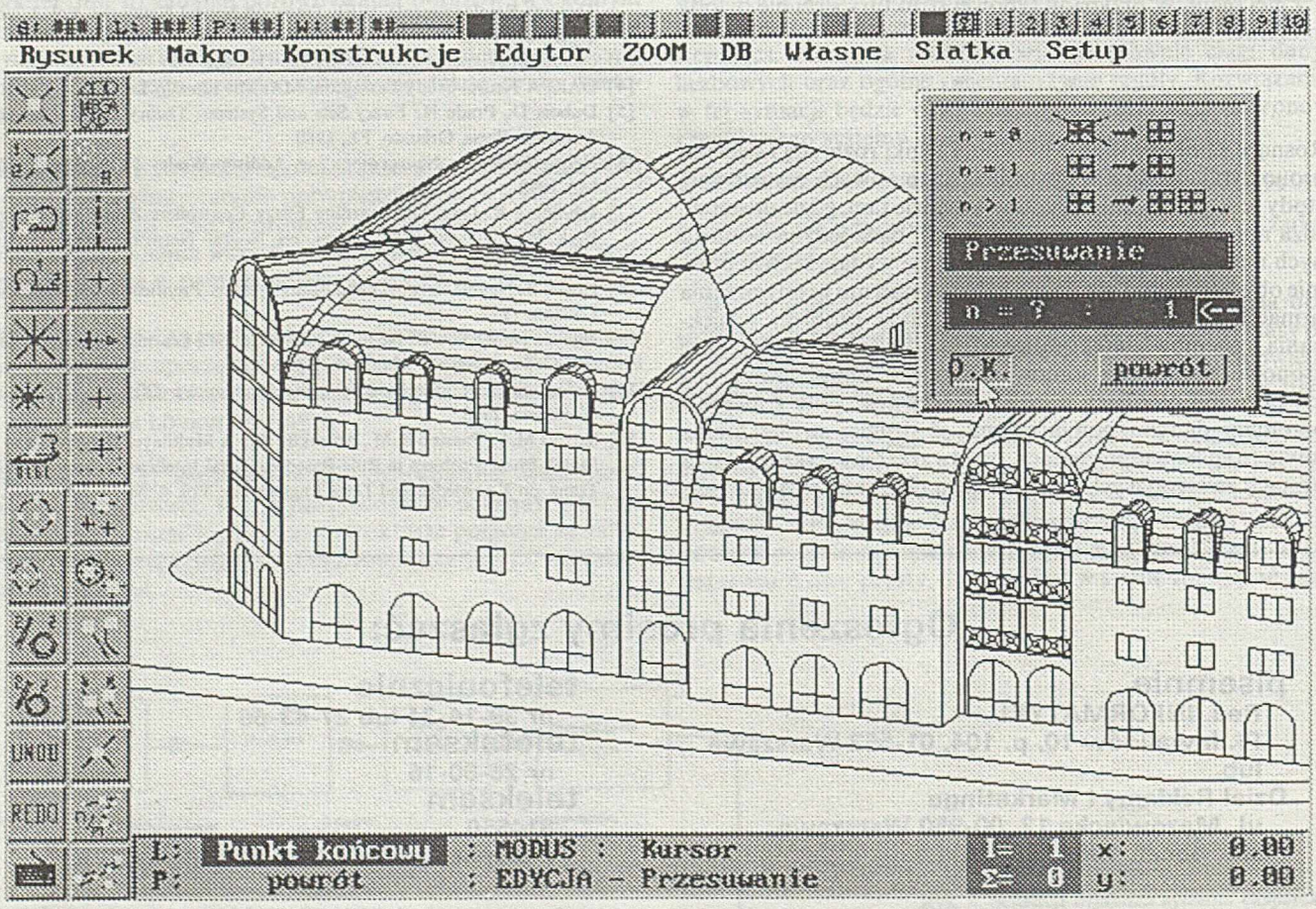
Zainteresowanych i wątpiących prosimy o kontakt z firmą CAD PO POLSKU w celu uzgodnienia terminu i miejsca prezentacji.

Generalny dystrybutor CAD PO POLSKU serdecznie zaprasza do swojego stoiska na targach CAD/CAM w Warszawie w dniach 7–10 czerwca br.

JOANNA METELKIN

CAD PO POLSKU, 01-494 Warszawa, ul. Sołtana 12/24

tel., faks: (0-2) 628-65-70



MAREK GAJĘCKI
KRZYSZTOF ZIELIŃSKI
Instytut Informatyki
Akademia Górniczo-Hutnicza
Kraków

PVM środowisko do przetwarzania rozproszonego w sieciach stacji roboczych

Parallel Virtual Machine PVM, system oprogramowania do realizacji rozproszonych programów w heterogenicznych sieciach maszyn sekwencyjnych, równoległych i wektorowych został opracowany w Oak Ridge National Laboratory, Tennessee, USA. Prace nad projektem rozpoczęto latem 1989 r. Od tego czasu projekt ten doskonalono i rozszerzano na nowe platformy sprzętowe. W lutym 1993 r. powstała wersja 3.0 tego pakietu.

PVM jest oprogramowaniem dostępnym nieodpłatnie, np. przez sieć interenet pod adresem cs.utk.edu.

Obecnie PVM jest dostępny na kilkunastu architekturach, od procesora Intel 386 poczynając, przez praktycznie wszystkie typy stacji roboczych, na architekturze CRAY kończąc.

Celem artykułu jest prezentacja jednego z narzędzi do programowania rozproszonego w sieci stacji roboczych oraz przekazanie wstępnych doświadczeń z jego eksploatacji.

Charakterystyka ogólna PVM

PVM jest narzędziem służącym do tworzenia programów rozproszonych, liczących się w sieci stacji roboczych. Wiele komputerów połączonych siecią tworzy tzw. równoległą maszynę wirtualną, na której można uruchomić program złożony z wielu wzajemnie komunikujących się procesów. Komputery wchodzące w skład maszyny wirtualnej mogą mieć różne architektury sprzętowe. PVM zapewnia automatyczną konwersję między różnymi reprezentacjami danych. Komputery, na których jest realizowane obliczenie rozproszone, mogą być połączone siecią zarówno lokalną, jak i rozległą. Daje to możliwość tworzenia „komputera wirtualnego" o ogromnej mocy obliczeniowej. Na każdym komputerze wchodzącym w skład maszyny wirtualnej musi zostać uruchomiony odpowiedni program – *demon*. Poprzez demony są uruchamiane i kontrolowane procesy użytkownika. Demony pośredniczą w komunikacji między procesami wchodzącymi w skład programu rozproszonego, a także zapewniają konwersję danych między różnymi architekturami. Możliwa jest także komunikacja z pominięciem demonów, co w określonych sytuacjach może spowodować jej przyspieszenie.

Z punktu widzenia programisty, PVM nie jest językiem programowania, ale zbiorem kilkunastu funkcji bibliotecznych, co stanowi o jego prostocie. PVM jest oparty na modelu obliczeń typu *distributed memory* oraz *message passing*. Głównym jego zadaniem jest dostarczenie funkcji do komunikacji między procesami. Oprócz funkcji komunikacyjnych PVM dostarcza wielu funkcji pomocniczych, np. zarządzania procesami, ich synchronizacją itp. Na uwagę zasługują funkcje pozwalające testować i rekonfigurować maszynę wirtualną z wnętrza programu. Umożliwia to tworzenie programów reagujących na zmiany w konfiguracji sieci, np. wyłączenie niektórych stacji roboczych biorących udział w obliczeniu rozproszonym nie musi powodować przerwania obliczeń.

W skład systemu PVM wchodzą:
- program konsoli służącej do zarządzania maszyną wirtualną – program pvm,
- biblioteka funkcji PVM – zbiór **libpvm3.a**,
- biblioteka z funkcjami operującymi na grupach procesów – zbiór **libgpvm3.a**,
- program demon umożliwiający współdziałanie poszczególnych programów (procesów UNIX-a) składających się na program rozproszony – zbiór **pvmd3.**

Po uruchomieniu konsola zgłasza się znakiem zachęty sygnalizując gotowość przyjmowania poleceń. Ważniejsze z poleceć to:
add – dodanie hosta (stacji roboczej) do maszyny wirtualnej,
delete – usunięcie hosta z maszyny wirtualnej,
conf – wypisanie maszyn wchodzących w skład maszyny wirtualnej,
reset – zakończenie wykonywania wszystkich procesów użytkownika,
quit – zakończenie pracy konsoli – maszyna wirtualna działa nadal,
halt – zakończenie pracy PVM – wszystkich procesów, demonów PVM i konsoli.

Przed uruchomieniem właściwego programu rozproszonego należy zainicjować maszynę wirtualną. Istnieje kilka sposobów konfigurowania i uruchamiania maszyny wirtualnej. Pierwszy sposób polega na uruchomieniu konsoli i ręcznym dodaniu poszczególnych stacji roboczych do maszyny wirtualnej. Drugim sposobem jest uruchomienie programu **pvmd3** z parametrem, którym jest plik tekstowy opisujący konfigurację. Trzeci sposób, to uruchomienie programu na jednej maszynie, a następnie za pomocą odpowiednich funkcji bibliotecznych konfiguracja maszyny wirtualnej i uruchomienie na niej odpowiednich zadań z wnętrza programu.

Na każdej maszynie wchodzącej w skład maszyny wirtualnej jest startowany program demon. Po ich wystartowaniu maszyna wirtualna jest gotowa do pracy. Do uruchamiania demonów na

kolejnych mszynach jest używany unixowy mechanizm rsh. Zatem warunkiem użycia stacji roboczej jako składowej PVM jest możliwość wykonywania na niej poleceń za pomocą mechanizmu *rsh*. Czasem w tym celu może być konieczne utworzenie odpowiednich zbiorów *.rhosts*.

PVM jest najczęściej uruchamiany na podstawie informacji zawartej w pliku konfiguracyjnym. W pliku tym jako pierwszą należy umieścić maszynę, z której będzie uruchamiany program. W następnych liniach pliku wprowadzamy nazwy pozostałych maszyn, na których będziemy realizować obliczenie rozproszone. W pliku konfiguracyjnym można umieścić dodatkowe informacje, np. ścieżki dostępu do demonów.

Przykład pliku konfiguracyjnego obejmującego trzy maszyny.

```
# nazwa hosta z programem master
oberon

# opcja ep wskazuje gdzie znajduje sie program typu worker
krak.cyf-kr.edu.pl ep=~/primes

# opcja dx wskazuje gdzie znajduje sie demon PVM
galaxy dx=/usr/local/bin/pvmd3
```

Po zakończeniu obliczeń należy zlikwidować demony (najwygodniej jest wykonać polecenie *halt* na konsoli PVM).

Funkcje biblioteczne PVM

Drugą część PVM stanowi biblioteka funkcji. W bibliotece PVM można wyróżnić kilka grup funkcji, których podział przedstawiono w dwóch tabelach. W pierwszej z nich znajdują się funkcje używane praktycznie zawsze w programie rozproszonym. Druga tabela zawiera funkcje używane rzadziej i używane w bardziej zaawansowanych programach. Podział ten jest podziałem orientacyjnym i subiektywnym, wynikającym z praktyki autora.

Biblioteka dostępna jest dla języka C i języka FORTRAN, ale nie wydaje się być problemem dołączenie jej funkcji do innego języka oprogramowania. Pakiet PVM jest dostarczany w postaci źródłowej (język C) i należy go skompilować na konkretnej architekturze sprzętowej (wydruk w sąsiedniej szpalcie).

Funkcje związane z buforami komunikacyjnymi stosuje się rzadko. W większości przypadków wystarcza jeden istniejący bufor. Funkcje zarządzania grupami zgromadzono w dodatkowej bibliotece **libgpvmm3.a**, która musi być dodatkowo dołączana na etapie kompilacji. Na wydruku 2 podano przykład funkcji pakującej i rozpakowującej dane typu *integer*. Istnieją oczywiście analogiczne funkcje dla pozostałych typów danych. Jeden przesyłany pakiet informacji może zawierać dane różnych typów, a także tablice.

Dokładne omówienie semantyki wszystkich funkcji przekracza ramy artykułu. Omówimy tylko funkcje przesyłania danych. Do wysłania danych do innego procesu służy funkcja *pvm_send*. W jej wywołaniu należy przekazać dwa parametry. Pierwszy, to identyfikator procesu odbiorcy (ang. *task identyfier*), drugi – numer przesyłanej wiadomości (ang. *message tag*). Numerowanie wiadomości pozwala np. selektywnie wybierać nadchodzące wiadomości. W przypadku funkcji *pvm_mcast* wysyłającej informacje od wielu procesów, przekazujemy w jej wywołaniu trzy parametry: tablicę zawierającą identyfikatory odbiorców, ich liczbę oraz numer rozsyłanej wiadomości. Należy wyjaśnić, że funkcja *pvm_mcast* wysyła do odbiorców wiadomości sekwencyjne, a nie jak mogłoby się wydawać, za pomocą sieciowego mechanizmu rozgłaszania (ang. *broadcast*)

Funkcje zarządzania procesami PVM	
pvm_mytid	rejestracja procesu w maszynie wirtualnej;
pvm_exit	wyrejestrowanie procesu z maszyny wirtualnej;
pvm_spawn	uruchomienie jednego lub więcej procesów;
Funkcje pomocnicze	
pvm_parent	zwraca numer identyfikacyjny procesu rodzica;
pvm_stat	zwraca status procesu;
pvm_mstat	zwraca status hosta;
pvm_config	pozwala odczytać konfigurację maszyny wirtualnej;
pvm_tasks	pozwala odczytać informacje o procesach PVM;
Funkcje komunikacyjne oraz funkcje pakowania i rozpakowywania danych	
pvm_initsend	przygotowanie bufora dla przesłania informacji;
pvm_send	wysłanie informacji;
pvm_mcast	wysłanie informacji do wielu procesów;
pvm_recv	odebranie informacji (zawieszające);
pvm_nrecv	odebranie informacji (niezawieszające);
pvm_pkint	zapakowanie danych typu integer do przesłania;
pvm_upkint	wypakowanie danych typu integer po przesłaniu;

Funkcje dynamicznej rekonfiguracji maszyny wirtualnej	
pvm_addhost	dodanie hosta do maszyny wirtualnej;
pvm_delhost	usunięcie hosta z maszyny wirtualnej;
Funkcje sygnalizacji między procesami	
pvm_sendsig	wysłanie sygnału do procesu;
pvm_notify	zawiadomienie procesów o zajściu zdarzenia;
Funkcje obsługi błędów	
pvm_perror	wypisuje status ostatnio wykonanej operacji PVM;
pvm_serror	załącza lub wyłącza reagowanie na błędy;
Funkcje związane z buforami komunikacyjnymi	
pvm_mkbuf	utworzenie nowego bufora do komunikacji;
pvm_freebuf	zwolnienie bufora;
pvm_getsbuf	pobranie aktualnego bufora dla wysyłania;
pvm_getrbuf	pobranie aktualnego bufora dla odbierania;
pvm_setsbuf	wybranie bufora dla nadawania;
pvm_setrbuf	wybranie bufora dla odbierania;
Funkcje zarządzania grupami (wybrane)	
pvm_joingroup	dołączenie procesu do grupy;
pvm_lvgroup	odłączenie procesu do grupy;
pvm_barrier	synchronizacja wszystkich członków grupy;
pvm_bcast	wysłanie komunikatu do członków grupy;

Wydruk

Odebrać informacje można na dwa sposoby:

- funkcją *pvm_recv*, która zawiesza wykonanie programu do czasu otrzymania właściwej wiadomości,
- funkcją *pvm_nrecv*, która nie zawiesza wykonywania programu, a przez zwracaną wartość sygnalizuje fakt odebrania lub nieodebrania wiadomości.

Drugiej z tych funkcji używamy wtedy, gdy nie potrafimy określić, kiedy przybędzie informacja i nie chcemy blokować procesu aktywnym oczekiwaniem. W przypadku obu funkcji odbierających, w wywołaniu przekazujemy dwa parametry: identyfikator procesu nadawcy oraz numer wiadomości na jaką oczekujemy. Każdy z tych parametrów może być zastąpiony wartością – 1. Sprawia to, że w pierwszym przypadku będą odbierane wiadomości od dowolnego procesu, w drugim – wiadomości o dowolnym numerze.

W wersji biblioteki dostępnej w języku FORTRAN istnieją niewielkie różnice przy przekazywaniu parametrów, wynikające z różnic w typach danych.

Technika tworzenia oprogramowania w środowisku PVM

Stworzony za pomocą PVM program rozproszony składa się z wielu wzajemnie komunikujących się programów (procesów

UNIX-a) lub kopii tego samego programu. Funkcje dostarczane przez PVM pozwalają tworzyć programy, w których występuje zarówno równoległość na poziomie kodu programu (różne procesory liczą inne fragmenty kodu), jak i równoległość na poziomie danych (wszystkie procesory wykonują ten sam kod dla różnych danych).

PVM nie zawiera narzędzi dla tworzenia architektur logicznych. Jak wynika z przedstawionego omówienia funkcji komunikacyjnych, komunikować może się każdy proces z każdym. Jest to zgodne z architekturą fizyczną (sieć komputerów połączonych wspólnym medium transmisyjnym). W przypadku, gdy z charakteru problemu wynika, że komunikacja powinna odbywać się zgodnie z inną architekturą (pierścień, krata, drzewo), użytkownik powinien sam zadbać o transformację adresów wynikających z architektury logicznej na odpowiednie adresy w funkcjach komunikacyjnych. Transformacja ta jest dokonywana w czasie tworzenia programu i pozostaje z nim związana.

W wielu problemach, które chcemy rozwiązać w sposób rozproszony, podział zadania jest dokonywany przez podział danych. Podział taki umożliwia rozwiązywanie problemu na różnej liczbie stacji roboczych (skalowanie problemu). Ze względu na prostotę, jak i często występujące skalowanie, częściej używa się sposobu obliczeń z równoległością danych, który występuje pod nazwą *MASTER-WORKER*.

Jeden z programów, nazywany dalej *MASTER* jest wyróżniony przez to, że uruchamiany jest jako pierwszy. W nim też najczęściej zawarta jest obsługa we-wy. Program ten powinien zarejestrować się wykonując funkcję *pvm_mytid*. Funkcja *pvm_mytid* zwraca numer identyfikujący proces *pid*. Numer ten, inny dla każdego procesu, używany jest przy funkcjach komunikacyjnych jako adres nadawcy lub odbiorcy. Program *MASTERA* uruchamia pewną liczbę innych procesów typu *WORKER* (funkcja *pvm_spawn*). Funkcja *pvm_spawn* dostarcza tablicę numerów uruchomionych procesów, umożliwiając *MASTER-owi* przesyłanie do nich danych. Maszynę, na której zostanie uruchomiony proces możemy wskazać przez parametr funkcji *pvm_spawn* lub pozostawić wybór PVM.

Na początku każdy program typu *WORKER* powinien zarejestrować się jako klient maszyny wirtualnej, wykonując funkcję *pvm_mytid*. Funkcja ta zwraca numer identyfikujący proces (identyczny z numerem, jaki zwróciła funkcja *pvm_spawn*) lub sygnalizuje błąd. Proces typu *WORKER*, aby wysłać informację do *MASTER-a* musi znać jego numer identyfikujący. Dowiedzieć się tego może wykonując funkcję *pvm_parent*. W przypadku, gdy niezbędna jest komunikacja między *WORKER-ami*, *MASTER* musi rozesłać tablicę zwróconą przez *pvm_spawn* do wszystkich procesów *WORKER*. Rozesłanie tablicy jest również niezbędne, gdy chcemy wprowadzić własne, łatwiejsze w użyciu, identyfikatory procesów, np. ponumerować procesy typu *WORKER* kolejnymi liczbami naturalnymi. *WORKER* określa swój numer kolejny, porównując numer otrzymany z funkcji *pvm_init* z jego pozycją w otrzymanej od *MASTER-a* tablicy.

Program typu *WORKER* może zakończyć się samodzielnie, wykonując kolejno funkcję *pvm_exit* – odłączenie od maszyny wirtualnej i *exit*. Może też zostać zakończony przez proces, który go uruchomił (funkcja *pvm_kill*). Program główny przed zakończeniem także powinien wykonać funkcję *pvm_exit*.

Często wygodnie jest umieścić program *MASTER-a* i *WORKER-a* w jednym zbiorze (szczególnie, gdy wykonują one podobny kod). Zmniejsza to rozmiar całego programu (biblioteka z funkcjami PVM jest dołączana tylko raz). Wspólny proces po wykonaniu funkcji *pvm_init* powinien wywołać funkcję *pvm_parent* i w zależności od jej wyniku wykonać funkcję *MASTER-a* albo *WORKER-a*. Możliwe jest też zintegrowanie programu sekwencyjnego z wersją rozproszoną.

```
mytid = pvm_mytid();
if (mytid<0) {  /* brak maszyny wirtualnej - wersja sekwencyjna */
  .
} else {
  parent = pvm_parent();
  if (parent<0) { /* kod MASTER-a */
    pvm_spawn(...); /* tworzenie proces\'ow typu WORKER */
    .
  } else { /* kod WORKER-a */
    .
  }
}
```

Przykład programu rozproszonego

Jako przykład rozważmy problem zaliczania w danym przedziale liczb spełniających określony warunek (np. liczb pierwszych).

Program sekwencyjny może mieć następującą postać:

```
#include <stdio.h>

main()
{
  int left,right,i,count;

  left = 1;
  right = 10001;

  count = 0;
  i = left;
  while (i<right)
    if (IsPrime(i++)) count++;

  printf( "Found %d primes\n",count );
}
```

W tym przypadku rozproszenie polegać będzie na podziale całego przedziału liczbowego na podprzedziały, dla których obliczenia zostaną wykonane równolegle na wielu stacjach roboczych. Ze względu na możliwą różną moc obliczeniową poszczególnych stacji, różne ich obciążenie, a także różny czas obliczania funkcji *IsPrime*, dla różnych wartości argumentu, nie można arbitralnie podzielić całego przedziału na równe części (ang. *static load balancing*), gdyż spowoduje to nierównomierne wykorzystanie procesorów i wydłużenie czasu obliczeń. Aby zapewnić właściwe wykorzystanie procesorów, cały przedział zostanie podzielony na dostatecznie wiele małych części, a podzadania będą przydzielane w trakcie wykonywania programu (ang. *dynamic load balancing*). Po zakończeniu obliczeń w danym przedziale *WORKER* odsyła wyniki, a następnie otrzymuje kolejne dane. Rozwiązanie to zapewnia właściwe zrównoważenie obciążenia procesorów (jego wadą jest większa liczba komunikacji w czasie obliczeń).

Używający PVM rozproszony program składa się z procesu *MASTER* i wielu procesów typu *WORKER*. Danymi dla każdego podzadania są początek i koniec sprawdzanego przedziału. Wynikiem jest ilość liczb pierwszych w przedziale.

Razem z wynikiem częściowym jest przesyłany numer identyfikujący *WORKER-a*. Informacja ta jest niezbędna, aby *MASTER* wysłał następne dane do właściwego *WORKER-a*. W programie tym brak jest buforowania danych w procesie

WORKER-a. Przyczyną jest sam PVM, który nie dostarcza takich możliwości. Brak buforowania jest przyczyną powstawania okresów bezczynności między momentem wysłania wyników częściowych, a otrzymaniem nowych danych.

Proces *MASTER-a*

```
#include <stdio.h>
#include "pvm3.h"

main()
{
  int mytid,master;
  int nhost,narch;
  int tids[32];
  int msgtype;
  int i,count,left,right,tid;
  int nproc;
  int g_count,g_left,g_right;
  int num_per_pack,total_pack;
  int snd_pack,rcv_pack;

  g_left = 1;
  g_right = 10001;              /* zakres poszukiwan */

  mytid = pvm_mytid();
  pvm_config( &nhost, &narch, NULL );
  printf("%d hosts found\n",nhost);

  nproc = nhost;
  pvm_spawn( "cpnw", (char**)0, 0, "", nproc, tids );

  g_count = 0;              /* globalny licznik znalezionych liczb */
  snd_pack = 0;             /* licznik wyslanych podzadan */
  rcv_pack = 0;             /* licznik odebranych wynikow */
  total_pack = 1000;        /* laczna ilosc podzadan */
  num_per_pack = (g_right-g_left)/total_pack;
                            /* wielkosc podzadania */

  /* wstepne rozeslanie zadan */
  for (i=0; i<nproc; i++) {
    left = g_left + num_per_pack*snd_pack;
    right = g_left + num_per_pack*(snd_pack+1);
    pvm_initsend( PvmDataDefault );
    pvm_pkint( &left, 1, 1 );
    pvm_pkint( &right, 1, 1 );
    msgtype = 200;
    pvm_send( tids[i],msgtype ); /* wyslanie danych dla podzadania */
    snd_pack++;
  }
  /* glowna petla rozsylajaca dane i odbierajaca wyniki */
  while (rcv_pack<total_pack) {
    msgtype = 201;
    pvm_recv( -1,msgtype );     /* odebranie wynikow */
    pvm_upkint( &count, 1, 1 );
    pvm_upkint( &tid, 1, 1 );
    rcv_pack++;
    g_count += count;
    if (snd_pack<total_pack) {
      left = g_left + num_per_pack*snd_pack;
      right = g_left + num_per_pack*(snd_pack+1);
      pvm_initsend( PvmDataDefault );
      pvm_pkint( &left, 1, 1 );
      pvm_pkint( &right, 1, 1 );
      msgtype = 200;
      pvm_send( tid,msgtype ); /* wyslanie kolejnych danych */
      snd_pack++;
    }
  }

  printf( "Found %d primes\n",g_count );

  /* zakonczenie pracy WORKER-ow */
  for (i=0; i<nproc; i++) pvm_kill( tids[i] );
  pvm_exit();
}
```

Proces *WORKER-a*

```
#include "pvm3.h"

main()
{
  int mytid,master;
  int msgtype;
  int i,count;
  int left,right;

  mytid = pvm_mytid();
  master = pvm_parent();

  while (1) {
    /* odbierz dane od MASTER-a */
    msgtype = 200;
    pvm_recv( master,msgtype );
    pvm_upkint( &left, 1, 1 );
    pvm_upkint( &right, 1, 1 );

    /* obliczenie dla podzadania */
    count = 0;
    i = left;
    while (i<right)
      if (IsPrime(i++)) count++;

    /* wyslanie wynikow do MASTER-a */
    pvm_initsend( PvmDataDefault );
    pvm_pkint( &count, 1, 1 );
    pvm_pkint( &mytid, 1, 1 );
    msgtype = 201;
    pvm_send( master,msgtype );
  }
}
```

Po wykonaniu przez proces MASTER funkcji *pvm_spawn* tablica *tids* zawiera numery identyfikujące poszczególne procesy typu *WORKER*. Numery te są niezbędne przy rozsyłaniu danych przez *MASTER-a*.

Zakończenie procesów typu *WORKER* jest realizowane przez proces *MASTER* funkcją *pvm_kill*.

W powyższym przykładzie procesy *MASTER* i *WORKER* są zawarte w oddzielnych plikach. Innym, czasem wygodniejszym rozwiązaniem, jest umieszczenie obu procesów w jednym pliku.

* * *

Zalety systemu PVM to:
- prostota (kilkanaście funkcji bibliotecznych),
- łatwość implementacji na różnych platformach sprzętowych,
- możliwość działania w sieci heterogenicznej,
- dowolny zasięg działania.

Do wad systemu PVM można zaliczyć brak wielowątkowości o obrębie procesu. Powoduje to, że trudno jest zrealizować np. buforowanie danych w modelu *MASTER-WORKER*.

PVM jest produktem wciąż rozwijającym się, jednak w obecnej wersji wydaje się być na tyle dojrzały, aby móc konkurować z innymi tego rodzaju narzędziami zwłaszcza, że jest dostępny dla wszystkich użytkowników. Do pierwszych eksperymentów wystarcza komputer klasy 386 z systemem operacyjnym Linux (wersja UNIX-a dostępna nieodpłatnie). Pewne problemy występują przy użyciu funkcji operujących grupami, jednak w większości programów można zrezygnować z użycia grup.

dokończenie na s. 28

Andrzej Potempa
ZETO-RODAN

System CAD/CAM/CAE UNIGRAPHICS

System UNIGRAPHICS jest własnością firmy EDS (Electronic Data Systems - General Motors). EDS jest światowym liderem w dostarczaniu systemów inżynierskich. Obrót finansowy na koniec 1993 roku przekroczył 8,56 miliarda dolarów amerykańskich. Firma zatrudnia ponad 70.000 pracowników w 30 krajach. Ponad siedem i pół tysiąca klientów na całym świecie używa systemów dostarczonych przez EDS.

UNIGRAPHICS jest systemem CAD/CAM/CAE stosowanym w procesach projektowo-konstrukcyjnych oraz do technologicznego przygotowania produkcji w tym obliczeń inżynierskich i analizie. Został stworzony na potrzeby znanej korporacji lotniczej McDonnell Douglas. Był rozwijany systematycznie przez trzydzieści lat dla potrzeb lotnictwa, jednak ze względu na ogromne możliwości szybko został wprowadzony w przemyśle samochodowym i maszynowym. W roku 1991 prawa do tego systemu przejął EDS, a na początku 1993 roku dostarczana jest na rynek udoskonalona wesja 10.

Wersja 10 systemu UNIGRAPHICS przełamuje definitywnie klasyczny podział na komputerowe projektowanie (CAD) i technologiczne przygotowanie produkcji (CAM), stając się systemem w pełni zintegrowanym, pozwalającym na jednoczesną pracę na wielu stanowiskach we wszystkich fazach projektu. Dzięki połączeniu z systemem INFOMANAGER cały cykl życia projektu wyrobu może być zarządzany i kontrolowany przez uczestników procesu, bez konieczności kosztownych weryfikacji i uzgodnień po każdej fazie tego projektu. Zastosowanie pełnej wersji 10 pozwoliło skrócić cykl projektu w nawet najbardziej zaawansowanych ośrodkach projektowych z kilkunastu do kilku miesięcy. Wszystkie moduły systemu UNIGRAPHICS korzystają z jednej wspólnej bazy danych 3D (trójwymiarowej). Wszelkie dane lub modyfikacje wprowadzane w poszczególnych modułach są automatycznie przekazywane do wszystkich pozostałych. Daje to możliwość jednoczesnego projektowania na wielu stanowiskach, co w sposób znaczący pozwala zredukować czas weryfikacji i kreślenia, jak również zwiększa dokładność i jakość projektowania. Ostatnia cecha systemu wiąże się z wprowadzaną powszechnie w zaawansowanych procesach przemysłowych praktyką Concurrent Engineering (Projektowanie Współbieżne). Ma to niewątpliwy wpływ nie tylko na wspomniane skrócenie czasu technicznego przygotowania produkcji wyrobu, lecz na nie notowany dotychczas wzrost poziomu jakości w dostosowaniu się do wysokich wymagań rynkowych.

System ma budowę modułową, a do najważniejszych modułów należy zaliczyć :

A. Moduły CAD (Computer Aided Design):
Modelowanie brył (Solids Modeling),
Modelowanie szczegółów brył (Feature Modeling),
Modelowanie powierzchni (Free-Form Modeling),
Modelowanie złożeń (Assembly Modeling),
Moduł kreślenia (Drafting).

b. Moduły CAM (Computer Aided Manufacturing):
Toczenie (Lathe).
Frezowanie (Planar Milling),
Frezowanie powierzchni (Surface Milling),
Frezowanie form (Cavity Milling),

C. Moduły CAM (Computer Aided Engineering):
Obliczenia metodą elementu brzegowego (TEAM),
Obliczenia metodą elementów skończonych (GFEM Plus),
Obliczenia rozkrojów na blachach (Sheet Metal),
Obliczenia kinematyczne mechanizmów (Mechanisms).

Oprogramowaniem ściśle współpracującym z systemem UNIGRAPHICS jest INFOMANAGER (IMAN).
Jest to system, który zapewnia kontrolowany i bezpieczny dostęp do wszystkich danych dotyczących konkretnego projektu przez wiele osób jednocześnie. Do najważniejszych funkcji systemu IMAN należą :

- pełna integracja modułów CAD, CAM i CAE,
- kontrola postępu prac przez upoważnione osoby,
- łatwość weryfikacji poszczególnych faz projektu wyrobu,
- pełna kontrola cyklu projektowania (zapis wszystkich wersji i zmian, odpowiedzialne osoby, etc),
- ustalenie zasad dostępu do danych dla poszczególnych konstruktorów i innych uczestników procesu przemysłowego,
- łatwość wykonywania zestawień materiałowych,
- pełna współzależność między zbiorami danych a strukturą wyrobu.

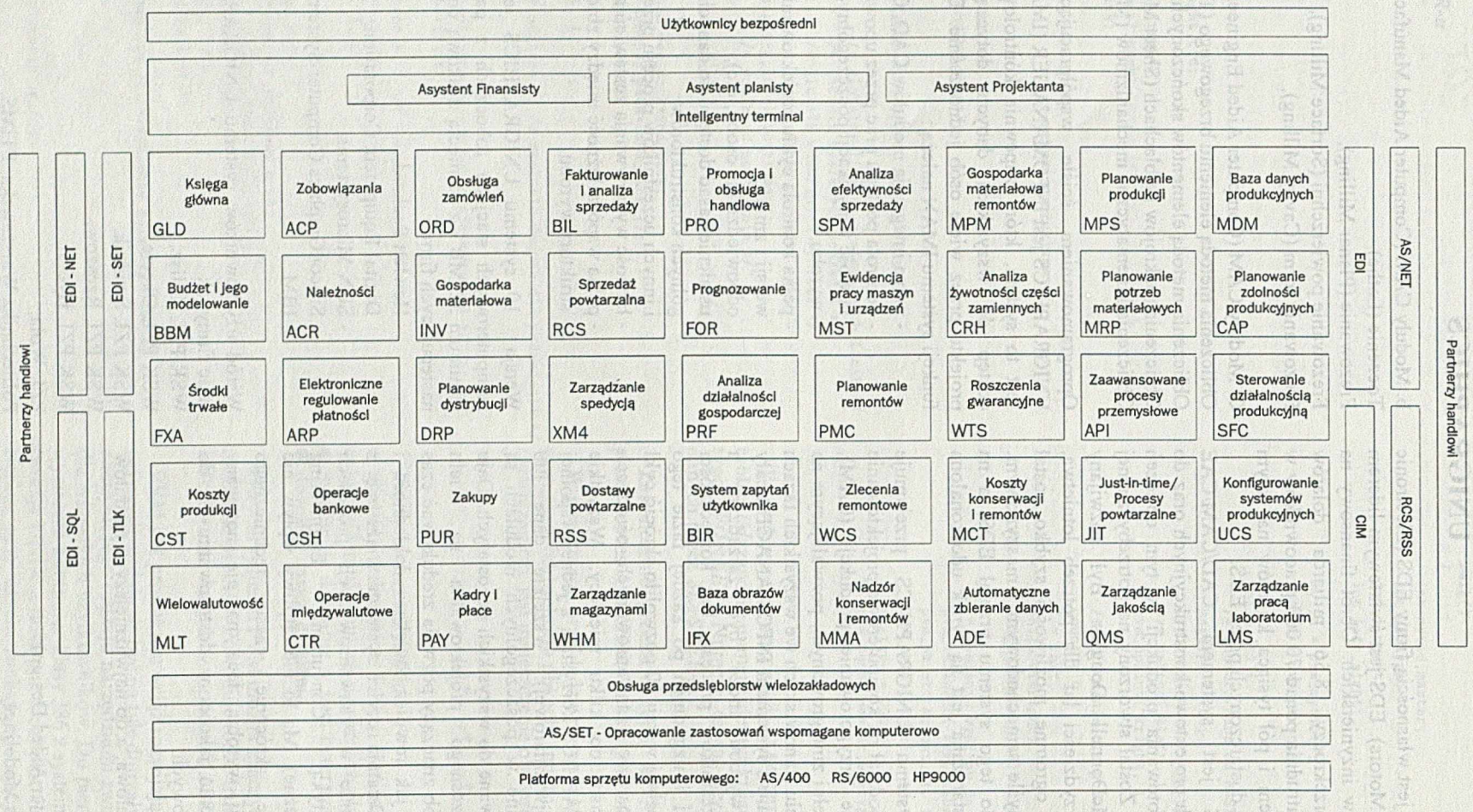
Wersja 10 systemu UNIGRAPHICS jest instalowana na komputerowych stacjach roboczych z pamięcią operacyjną minimum 32 MB i z pamięcią dyskową co najmniej 512 MB następujących firm:

- Hewlett Packard,
- Digital Equipment Corporation,
- SUN Microsystems,
- SiliconGraphics Computer Systems,
- IBM.

Wśród użytkowników systemu UNIGRAPHICS w Polsce są takie firmy jak :
WSK PZL Kalisz,
WSK PZL Okęcie,
WSK PZL Świdnik,
WSK PZL Rzeszów,
WSK PZL Hydral,
PZL Wola,
Politechnika Warszawska FTiMS,
Politechnika Warszawska MEiL,
DWLKK.

O/6/94

BPCS/AS™ : Otwarty system zarządzania przedsiębiorstwem



IDOM Poland
00-950 Warszawa
ul. Podwale 15

Tel./fax; 635 81 27
31 52 01
31 65 70

Oddziały: Wrocław
Poznań
Gdańsk

BPCS
zintegrowany system zarządzania

Zachodzące obecnie w Polsce, zakrojone na szeroką skalę procesy przekształcania gospodarki centralnie planowanej w gospodarkę rynkową w zasadniczy sposób zmieniły filozofię zarządzania przedsiębiorstwami. Niezbędnym stało się wprowadzenie do przedsiębiorstw wielu zmian, zarówno w strukturze organizacyjnej, jak i metodach zarządzania i finansowania ich działalności.

Fundamentalnym warunkiem sprawnego działania każdego przedsiębiorstwa, stała się szybka precyzyjna i wyczerpująca informacja, o ściśle określonym zakresie, niezbędnym dla poszczególnych poziomów zarządzania jego działalnością.

Funkcje te spełnić może jedynie spójny system informacyjny, umożliwiający bezkolizyjną wymianę informacji pomiędzy wszystkimi sferami działalności przedsiębiorstwa, tzn. począwszy od planowania i zaopatrzenia, przez produkcję do dystrybucji i sprzedaży, z równoległym kontrolowaniem na różnych etapach wyników finansowych tej działalności.

Wychodząc naprzeciw potrzebom polskiego rynku warszawska firma IDOM zajęła się dystrybucją i wdrażaniem systemu Business Planning and Control System, który cieszy się niekłamanym powodzeniem wśród najlepszych firm światowych, notowanych w pierwszej pięćsetce pisma „Fortune".

BPCS jest produktem amerykańskiej firmy SSA (System Software Associates) mającym obecnie ponad 7000 wdrożeń w 66 krajach, w tym również w Polsce. System obejmuje swoim zasięgiem takie obszary działania przedsiębiorstwa, jak produkcja i jej planowanie, dystrybucja, logistyka oraz gospodarka finansowa. Działa na podstawie systemu operacyjnego OS/400, który jest wewnętrznym systemem komputera AS/400, jak również jest dostępny dla pracujących pod kontrolą systemu UNIX komputerów HP 9000 i RS/6000. Zawiera ponad 45 zintegrowanych produktów programowych tzw. modułów. Modułowa budowa umożliwia w procesie wdrażania dobór tych składników aplikacji, które z uwagi na charakter przedsiębiorstwa i specyfikę jego działalności są niezbędne. Możliwe jest również wykorzystanie użytkowanego już oprogramowania i zintegrowanie go z systemem BPCS. Integracja systemu oznacza, że wszystkie moduły korzystają ze wspólnej bazy danych i wszelkie informacje zarejestrowane w jednym z nich są natychmiast dostępne dla użytkowników pozostałych modułów. Dzięki tej integracji wszystkie informacje wprowadzane są do systemu tylko raz, w miejscu ich powstania. Konsekwencją tego jest radykalne ograniczenie obiegu dokumentów w przedsiębiorstwie, zmniejszenie nakładu pracy oraz możliwość przechowywania danych bez zbędnej nadmierności. Zapewnia to spójność i aktualność danych.

Dostęp do zgromadzonej i przetworzonej informacji jest możliwy nie tylko z modułów BPCS, lecz również za pośrednictwem produktów pracujących na podłączonych do systemu inteligentnych stacjach roboczych. Obecnie oferowane programy oparte na architekturze klient–serwer to Asystent finansisty, Asystent planisty i Asystent projektanta. Pozwalają one na przeglądanie danych w różnych projektach, w środowisku graficznym oferowanym przez stację roboczą a także na prowadzenie różnych analiz.

Jedną z najmocniejszych stron systemu jest grupa modułów związanych z planowaniem. BPCS oprócz tradycyjnych metod ukierunkowanych głównie na planowanie potrzeb materiałowych (MRP) i zasobów produkcyjnych oferuje również moduły zarządzania jakością, remontami, pracą laboratorium, magazynami i innymi działami. W sumie system spełnia wymagania modelu określanego jako planowanie zasobów przedsiębiorstwa – ERP (Enterprise Resource Planning).

Stosowane metody planowania pozwalają na optymalizację wykorzystania różnorodnych zasobów przedsiębiorstwa, jak również na szybsze i bardziej elastyczne reagowanie na zmieniające się potrzeby rynku. System również wspiera wprowadzanie procedur kontroli jakości zgodnych z międzynarodowymi standardami, pozwalając przedsiębiorstwu ubiegać się o certyfikat zgodności z normami ISO 9000.

BPCS zapewnia obsługę różnych standardów elektronicznej wymiany danych (EDI), zaś wbudowane interfejsy pozwalają – przykładowo – na automatyczne przyjmowanie zamówień od klienta, potwierdzanie gotowości dostaw, czy też automatyczne fakturowanie.

Efektywne wdrożenie systemu BPCS wymaga określonego procesu zarządzania projektem. Dlatego dla potrzeb BPCS opracowano specjalną metodologię BASIS (BPCS Advanced System Implementation Strategy). Jest to w pełni sformalizowany i opisany sposób postępowania obejmujący wszystkie fazy wdrażania, od zdefiniowania projektu aż do optymalizacji działającego już systemu. Metodologia ta została opracowana na podstawie zbiorowego doświadczenia firmy SSA przy wdrażaniu około 7000 systemów informacyjnych dla przedsiębiorstw różnej wielkości, reprezentujących różny zakres działania.

Na strategię wdrożenia zaawansowanych systemów składa się pięć kluczowych elementów:

- *Proces opracowania projektu: analiza przedsiębiorstwa oraz definicja projektu.*
- *Organizacja prac projektowych: struktura i strategia działania zespołu ludzi potrzebnych do wdrożenia systemu, szkolenie, prototypowanie.*
- *Oprogramowanie i narzędzia: właściwe wykorzystanie systemu BPCS, mające na celu danie przedsiębiorstwu przewagi konkurencyjnej na rynku, zatwierdzenie zmian i modyfikacji.*
- *Przygotowanie do uruchomienia systemu.*
- *Optymalizacja systemu.*

Poza określoną metodologią planowania, organizacji, sterowania i kontrolowania przebiegu wdrożenia, istotną cechą BASIS-u jest fakt, że system wdrażany jest przez przyszłych użytkowników. Dzięki temu następuje szybki transfer wiedzy o systemie, co w połączeniu z „przyjazną" dla użytkownika budową systemu czyni, że mogą go obsługiwać osoby bez specjalnego przygotowania informatycznego, po krótkim szkoleniu.

Obecnie IDOM prowadzi wdrożenia systemu BPCS w STOMILU OLSZTYN S.A., CPC Amino, fabryce Thompson Polkolor, Bydgoskiej Fabryce Mebli oraz w Ferrero Polska Sp. z o.o.

IDOM POLAND

Licencjonowanym partnerem handlowym SSA w krajach Europy Środkowo-Wschodniej jest firma IDOM z siedzibą w Szwajcarii, posiadająca swe oddziały m.in. w Polsce, Czechach, Bułgarii i na Węgrzech. W ramach swej działalności IDOM oferuje pełny zakres usług, począwszy od szkoleń, przez zakup licencji BPCS, jej wdrożenie, dostawę wyposażenia komputerowego, projekty i instalacje techniczne sprzętu oraz nadzór pogwarancyjny dla wdrożonych projektów i licencji.

IDOM POLAND — **tel., faks: 635-81-27**
00-950 Warszawa — **31-52-01**
ul. Podwale 15 — **31-65-70**

O/5/94

ANNA ŁAWRYNOWICZ
Akademia Techniczno-Rolnicza
Bydoszcz

Ekspercki system wspomagania sterowania produkcją

Modele reprezentacji wiedzy, które powstają w nurcie badań nad sztuczną inteligencją stwarzają nowe możliwości w zakresie budowy komputerowych systemów wspomagających planowanie i sterowanie przebiegiem produkcji [2, 6]. Wraz z rozwojem tych modeli i sprzętu komputerowego powstają nowe urządzenia programowania. Jednym z narzędzi używanych w sztucznej inteligencji jest oparty na logice predykatów język Prolog [4, 8].

Techniki sztucznej inteligencji, a w szczególności systemy eksperckie, wydają się być bardzo przydatne do sterowania kompleksami operacji w dyskretnych procesach produkcyjnych [1, 7]. Opierając się na osiągnięciach w zakresie modelowania wiedzy i możliwościach w tym zakresie języka Turbo Prolog 2.0 (zaimplementowanego na komputerze IBM AT 386) zrealizowano ekspercki system wspomagania sterowania produkcją.

Sformułowanie problemu

Problematyka projektowania i sterowania kompleksami operacji z wykorzystaniem systemów eksperckich jest wspólna dla operacji o różnej naturze i obejmuje na ogół problemy rozdziału zadań i zasobów, szeregowania zadań oraz sterowania wykonaniem oddzielnych operacji w systemie [3]. Pojęcie kompleksu operacji odniesiono tu do dyskretnego procesu produkcyjnego, typowego dla przemysłu maszynowego, w którym najważniejszymi elementami są:

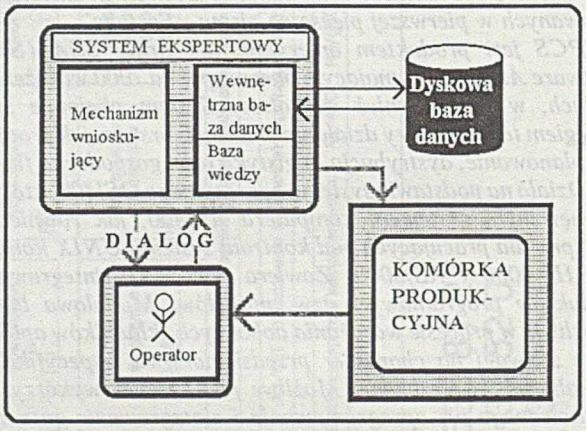
- zasoby (obrabiarki, detale ludzie),
- czasy operacji technologicznych,
- powiązania między operacjami technologicznymi, tj. struktura kompleksu oparta na określonych uwarunkowaniach czasowych (np. niektóre operacje mogą rozpocząć się dopiero po zakończeniu innych),
- priorytety będące miarą pilności wykonania detalu.

W przypadku procesów produkcyjnych, realizowanych m.in. w gniazdach o szybko zmiennym asortymencie produkcji, sterowanie kompleksami operacji staje się trudnym problemem dla dyspozytora systemu. Podejmowane w tej fazie produkcji niewłaściwe decyzje powodują na ogół wydłużenie czasów oczekiwania na wykonanie poszczególnych operacji, przeciążenie niektórych grup maszyn oraz niepełne wykorzystanie innych, co w rezultacie doprowadza do przekroczenia terminów wykonania detali. W związku z tym ciągle doskonalone są systemy komputerowe, których zadaniem jest wspomaganie pracy operatora koordynującego procesy realizacji zadań w systemie produkcyjnym.

Prezentowany system ekspercki wspomaga użytkownika w rozdziale operacji technologicznych ze skończonego ich zbioru do wykonania, na określonych stanowiskach obróbczych z uwzględnieniem następujących ograniczeń:

- operacje technologiczne na detalach, które mogą być przekazywane partiami lub pojedynczo ze stanowiska na stanowisko, są wykonywane bez przerwań (z wyjątkiem sytuacji, kiedy nastąpi awaria maszyny),
- stanowisko obróbcze w dowolnej chwili może wykonywać nie więcej niż jedną operację technologiczną,
- kolejność wykonywania operacji technologicznych na detalu jest ściśle określona,
- w pierwszej kolejności są wykonywane detale o minimalnej wartości wskaźnika priorytetu,
- ograniczona jest dostępność zasobów.

Na załączonym rysunku przedstawiono strukturę systemu wspomagającego użytkownika w sterowaniu produkcją.



Struktura systemu wspomagania sterowania produkcją

Sterowanie produkcją jest realizowane z wykorzystaniem następujących baz danych

- dzienny plan zadań komórki produkcyjnej,
- wykaz wolnych stanowisk obróbczych,
- wykaz pilnych zadań,
- zestawienie produkcji w toku,
- wykonanie planu zadań, a także pomocniczych baz danych, służących wyłącznie do przechowywania wyników pośrednich:
- wykaz zajętych stanowisk obróbczych,
- wykaz oczekujących na przydział zadań.

Wszystkie operacje na bazach danych są dokonywane w pamięci operacyjnej za pomocą predykatów dynamicznej zmiany pamięci.

Prezentowany system ekspercki jest systemem konwersacyjnym i w przypadku wystąpienia dużych zakłóceń w przebiegu procesu produkcyjnego operator może, za pośrednictwem klawiatury, zmieniać zawartość wszystkich baz danych.

Sterowanie produkcją jest realizowane z wykorzystaniem reguł „jeżeli/to" (*if/then*), które operują na faktach zawartych

w pamięci operacyjnej. Np. typowanie detali do uruchomienia (tworzenie zleceń) następuje z wykorzystaniem reguły:

Jeżeli **priorytet** ⟨wartość⟩ = ⟨minimum⟩
to **detal** ⟨kod⟩ **jest w stanie** ⟨oczekuje⟩.

Uruchamianie zleceń wykonania operacji technologicznych na detalach jest możliwe wtedy, gdy jest spełniona część warunkowa reguły:

Jeżeli **detal** ⟨kod⟩ **jest w stanie** ⟨oczekuje⟩
i **operacja** ⟨numer⟩ = ⟨minimum⟩
i **stanowisko** ⟨symbol⟩ **jest w stanie** ⟨wolne⟩
to URUCHOMIENIE
lub **detal** ⟨kod⟩ **jest w stanie** ⟨pilny⟩.

Opierając się na pierwszej regule z listy detali przewidzianych do wykonania, w systemie produkcyjnym zostaje wybrany ten, który ma najniższy wskaźnik priorytetu. Druga reguła z listy operacji przewidzianych do wykonania na tym detalu wybiera tę, która ma najniższy numer. W ten sposób jednoznacznie zostaje określona detalo-operacja. Jeżeli stanowisko, do którego została przypisana operacja jest wolne, to następuje uruchomienie zlecenia wykonania tej detalo-operacji. Ze względu na zmianę stanów zasobów uczestniczących w procesie produkcji, z uruchomieniem zlecenia wiąże się każdorazowo automatyczna aktualizacja baz danych.

Jeżeli po wytypowaniu detalo-operacji do wykonania okaże się, że stanowisko, do którego została ona przypisana jest zajęte, to wszystkie operacje do wykonania na wybranym detalu zostają umieszczone w bazie danych jako pilne. Po zwolnieniu jakiegokolwiek stanowiska, każdorazowo jest podejmowana próba uruchomienia w pierwszej kolejności pilnych zadań (z możliwością podwyższania wskaźnika priorytetu).

Realizacja programowa

Kolejne opcje głównego menu programu pozwalają użytkownikowi na:
- zakładanie, aktualizowanie i wyprowadzanie zawartości „wewnętrznych" baz danych,
- tworzenie zleceń,
- uruchamianie zleceń,
- kontrolę i ewidencję zleceń,
- przechowywanie zawartości baz danych w plikach (w pamięci dyskowej).

Aktywowanie wybranej z głównego menu opcji, związanej z zakładaniem lub utrzymywaniem baz danych, powoduje rozwinięcie podrzędnego menu w celu dokonania wyboru bazy danych, której zawartość (fakty) chcemy poznać lub modyfikować (np. w przypadku bardzo dużych zakłóceń w procesie produkcyjnym, wymagających ingerencji operatora systemu).

Tworzenie zleceń oraz typowanie detalo-operacji do uruchomienia jest realizowane z wykorzystaniem predykatu standardowego *findall*, który tworzy listę wszystkich rozwiązań zwracanych przez predykaty niedeterministyczne. Predykat ten ma następującą postać:

findall (*ZmiennaX*, *cel* (..., *ZmiennaX*,...), *Lista*)

i powoduje:
- wywołanie celu podanego jako drugi parametr,
- znalezienie wszystkich rozwiązań zmiennej wymienionej jako parametr pierwszy,
- utworzenie z tych rozwiązań listy, której nazwa jest podana jako parametr trzeci.

Przytoczony niżej fragment programu (wydruk 1) pokazuje użycie predykatu *findall*.

```
findall(Priorytet,db_planZad(_,_,Priorytet,_,_,_,_),ListaP),
minP(ListaP,MinimumP),
db_planZad(KodDet,NrOper,Priorytet,NazwaSt,Partia,CzasW,plan),
Priorytet=MinimumP,
assert(db_oczekuje(KodDet,NrOper,Priorytet,NazwaSt,Partia,
CzasW,oczekuje)),fail.
```

Wydruk 1

Predykat *minP* (*Listap*, *MinimumP*) podaje wartość najmniejszego elementu (*MinimumP*) listy (*ListaP*), a tym samym dokonuje wyboru detalu o najniższym wskaźniku priorytetu. Predykat standardowy *assert* powoduje umieszczenie operacji dla tego detalu w bazie danych detali oczekujących na obróbkę.

```
minP([G|[ ]],G).
minP([G|O],MinimumP) :-
minP(O,N1),
mi(G,N1,MinimumP),!.
mi(A,B,A) :-
A<B.
mi(A,B,B) :-
A>=B.
```

Wydruk 2

Sposób wybierania najmniejszego elementu z listy pokazano na wydruku 2.

* * *

Ogromną zaletą prologowych systemów eksperckich wspomagania planowania i sterowania produkcją, jak się wydaje, jest stosunkowo łatwe zarządzanie bazami danych [5]. Rozwiązuje to problem m.in. przygotowywania danych do podejmowania decyzji, co często stanowiło pewną barierę w rozwiązywaniu problemów dotyczących sterowania kompleksami operacji [9]. Struktura bazy wiedzy prezentowanego systemu eksperckiego umożliwia integrację sterowania na poziomie gniazda produkcyjnego z wyższymi szczeblami zarządzania produkcją. Dzienne plany zadań, które są budowane opierając się na bilansowaniu zadań z możliwościami produkcyjnymi na wyższych szczeblach zarządzania, mogą być przekazywane automatycznie do komórki produkcyjnej, której sterowanie wspomaga system ekspercki (za pośrednictwem pamięci dyskowej). Struktura bazy danych i bazy wiedzy umożliwia takie przekazywanie za pośrednictwem pamięci dyskowej stanu zaawansowania produkcji z dnia na dzień lub ze zmiany na zmianę i ułatwia aktualizację dziennych planów zadań, co istotnie wpływa na usprawnienie koordynacji procesu produkcyjnego.

LITERATURA

[1] Appel S., Ayel J., Laurent J., Lanusse A., Panet B.: Controller module for coordination of supervision activities on production management systems. Proceeding of the 5th CIM Europe Conference „Computer Integrated Manufacturing", Athens, 1989
[2] Bruno G., Elia A., Laface P., A rule-based system to schedule production. Artificial Intelligence in Industry. Spronger Verlag, 1988
[3] Bubnicki Z.: Wstęp do systemów ekspertowych. PWN, Warszawa, 1990
[4] Clark K.L., McCabe F.G.: Micro-Prolog. WNT, Warszawa, 1988
[5] Lucas R.: Database Applications Usig Prolog. Ellis Horwwood Limited, Chichester, 1988
[6] Ławrynowicz A.: Reprezentacja wiedzy w sztucznej inteligencji. Przegląd Mechaniczny 4, 1993
[7] Ławrynowicz A.: Sztuczna inteligencja w zarządzaniu produkcją. Przegląd Mechaniczny 9, 1993
[8] Merritt D..: Building Expert System in Prolog. Springer Verlag, 1989
[9] Zaborowski M.: Optymalizacja planów w systemach operatywnego sterowania produkcją. Archiwum Automatyki i Robotyki 2, 1991

PVM – środowisko do przetwarzania rozproszonego

dokończenie ze s. 22

Szybkość komunikacji w PVM przewyższa wszystkie do tej pory opisane w poprzednich numerach INFORMATYKI narzędzia do programowania rozproszonego. Na wydruku 7 podano czas przesyłania 1 KB danych między dwoma stacjami roboczymi SPARC Station SLC przy użyciu różnych narzędzi.

Narzędzie	t [msek]
Strand	30.23
Emerald	18.03
SR	10.20
PVM 2.4	9.51
Linda	6.57
ANSA	6.13
PVM 3.1	5.36

Większość dużych firm, takich jak Cray Research, Convex, SGI, IBM, Intel, czy Thinking Machines zamierza włączyć PVM do swoich systemów. Wszystko więc wskazuje, że PVM stanie się standardem w obliczeniach rozproszonych. Dostępna jest lista dyskusyjna *comp.parallel.pvm*, z której można uzyskać odpowiedzi na napotkane problemy oraz znaleźć przykłady zastosowań. System PVM jest używany przez studentów informatyki AGH podczas kursu systemów równoległych i rozproszonych.

LITERATURA

[1] Cap C.H., Strumpen V.: High Performance Platform for Parallel Computing in a Distributed Workstation Environment. Institut für Informatik, Univ. Zurich, 1992, TR IFI-92-07

[2] Cohn D.L., Delaney W.P., Tracey M.: Platform for Heterogeneous Distributed Operating Systems. Proc. of the Usenix Distributed and Multiprocessor Systems Workshop, 1989

[3] Douglas G.C., Mattson T.G., Schultz M.H.: Parallel Programming Systems for Workstation Clusters. Yale Univ. Dept. of Comp, S.C., August, 1993, RR YALEU/DCS/TR-975

[4] Gajęcki M., Jędrzejek K., Zieliński K.: Parallelization of the DBW Render Package on a Trasputer Farm. Machine GRAPHICS & VISION, Vol. 1, No. 4, 1993

[5] Kobielus J.: LANs Bid for Distributed Power. Network World, Dec. 3, 1990

[6] PVM 3.0 User's guide and reference manual

[7] Zieliński K., Gajęcki M., Czajkowski G.: Software Environments for LAN Distributed Computing – A Comparison Study. Institute of Computer Science, University of Mining and Metallurgy, TR 1993.

Przypominamy, że można zamówić prenumeratę naszego czasopisma na każdej poczcie w całym kraju oraz u doręczycieli, bez dodatkowej opłaty.

Kwaśnicka H., Markowska-Kaczmar U.: Sztuczna inteligencja w sterowaniu INFORMATYKA 1994, nr 6, s. 3 Zastosowanie metod logiki rozmytej do automatyzacji procesów, które zapewniają zwiększenie precyzji ich sterowania.	**Kwaśnicka H., Markowska-Kaczmar U.: Artificial Intelligence in process control** INFORMATYKA 1994, No. 6, p. 3 Application of fuzzy logic methods for process automatization, which secure precision increasing of its control.	**Kwaśnicka H., Markowska-Kaczmar U.: Künstliche Intelligenz in Steuerung** INFORMATYKA 1994, Nr. 6, S. 3 Anwendung von Fuzzy-Logic-Methoden für Prozessautomatisierung, die die Präzisionssteigerung der Prozesssteuerung sichert.
Grzegorzewski M., Rzesiowski D.: Sieci neuronowe i logika rozmyta w systemach eksperckich INFORMATYKA 1994, nr 6, s. 11 Wykorzystanie oraz łączenie metod zastosowania sieci neuronowych (neurokomputingu) i logiki rozmytej na przykładzie systemu eksperckiego.	**Grzegorzewski M., Rzesiowski D.: Neural networks and fuzzy logic in expert systems** INFORMATYKA 1994, No. 6, p. 11 Using and connecting of neural networks application (neurocomputing) and fuzzy logic methods on example of an expert system.	**Grzegorzewski M., Rzesiowski D.: Neuronennetze und Fuzzy Logic in Expertensystemen** INFORMATYKA 1994, Nr. 6, S. 11 Auswerten und Vereinen von Methoden der Neuronennetzeanwendung (Neurocomputing) und Fuzzy Logic auf Beispiel eines Expertensystems.
Gajęcki M., Zieliński K.: PVM – środowisko do przetwarzania rozproszonego w sieciach stacji roboczych INFORMATYKA 1994, nr 6, s. 19 Charakterystyka dostępnego nieodpłatnie amerykańskiego systemu oprogramowania PVM (*Parallel Virtual Machine*) do realizacji przetwarzania rozproszonego w sieciach stacji roboczych.	**Gajęcki M., Zieliński K.: PVM – the distributed processing environment in workstation networks** INFORMATYKA 1994, No. 6, p. 19 Characteristics of the unpaid american PVM (*Parallel Virtual Machine*) software system for distributed processing in workstation networks.	**Gajęcki M., Zieliński K.: PVM – eine Umwelt für zerstreute Datenverarbeitung** INFORMATYKA 1994, Nr. 6, S. 19 Eine Charakteristik von dem kostenlos erreichbaren amerikanischen PVM (*Parallel Virtual Machine*) Softwaresystem zur Realisierung der zerstreuten Datenverarbeitung in Workstationnetzen.
Ławrynowicz A.: Ekspercki system wspomagania sterowania produkcją INFORMATYKA 1994, nr 6, s. 26 Charakterystyka zrealizowanego na komputerze klasy 386 systemu eksperckiego wspomagania sterowania dyskretnym procesem produkcji.	**Ławrynowicz A.: An expert system for assisted manufacturing control** INFORMATYKA 1994, No. 6, p. 26 Characteristics of the, realized on 386 computer class, expert system for assisted controlling of discrete manufacturing process.	**Ławrynowicz A.: Ein Expertensystem zur Unterstützung von Produktionssteuerung** INFORMATYKA 1994, Nr. 6, S. 26 Eine Charakteristik von einem auf Computer der 386-Klasse realisierten Expertensystem für Steuerungsunterstützung eines diskreten Produktionsprozesses.

LOG|iC EVAL I LOG|iC LIGHT

Niemiecka firma ISDATA powiększyła niedawno swoją rodzinę programów LOG|iC projektujących układy PLD o dwa systemy dla MS-Windows: LOG|iC EVAL i LOG|iC LIGHT.

LOG|iC LIGHT to idealne rozwiązanie dla osób po raz pierwszy stykających się z układami PLD. Umożliwia realizację projektu od etapu idei do gotowych zbiorów dla programatora. Dysponuje własnym językiem programowania o strukturze blokowej, umożliwiającym opisanie projektu za pomocą równań logicznych, tablic prawdy lub tablic przejść automatów. Kompilator po sprawdzeniu składni bada spójność projektu a następnie przechodzi do etapu minimalizacji, wykorzystując jeden z trzech algorytmów: FACT, BRUNO lub Espresso. Etap ten pozwala przy większych projektach na znaczne oszczędności – układ mieszczący się początkowo w dwóch strukturach PLD po optymalizacji może zmieścić się w jednej!

Gdy mamy już skompilowany projekt możemy przystąpić do wyboru struktury, na której chcemy go zrealizować. LOG|iC ma bazę danych struktur dostępnych na rynku zawierającą wszelkie ich parametry potrzebne projektantowi (wersja LIGHT zawiera informacje o układach mających obudowy co najwyżej 24-nóżkowe). Po wybraniu układu musimy jeszcze uruchomić generator zbiorów wyjściowych ... i projekt gotowy!

Wersja LOG|iC EVAL, funkcjonalny odpowiednik wersji LIGHT, ma bazę danych ograniczoną do trzech rodzin układów PLD: 16V8, 20V8 i 22V10.

Świetnym uzupełnieniem LOG|iC-a jest nowy programator SPRINT PLUS 48 firmy SMS. Urządzenie to dołączane do równoległego portu drukarki programuje m.in. układy PLD i EPROM, stosując algorytmy zalecane przez producentów tych układów. Dzięki temu mamy zawsze gwarancję poprawnego zaprogramowania i sprawnego działania układu.

Dystrybucja: CADEL Sp. z o.o.

MegaCAD – prosty program dla wymagających

Program MegaCAD jest profesjonalnym systemem wspomagającym kreślenie rysunków technicznych. Program został stworzony przez niemiecką firmę MEGATECH i szybko stał się najbardziej popularnym systemem CAD na tym rynku. O popularności MegaCAD-a zadecydowały jego cztery główne zalety:

- *duża szybkość przy braku wymagań sprzętowych,*
- *duża łatwość i wygoda obsługi, krótki okres szkolenia,*
- *profesjonalne możliwości, zaawansowane funkcje edycji,*
- *konkurencyjna, niska cena.*

Największy nacisk przy tworzeniu programu położono na łatwość posługiwania się programem, zwłaszcza przez osoby, które wcześniej nie posługiwały się w pracy komputerem. Wszystkie funkcje są dostępne przez wybór ikony symbolizującej daną czynność. W trakcie realizacji funkcji rysowania elementu można zmieniać dynamicznie jego atrybuty, takie jak: rodzaj linii, styl, kolor itp. Program ma rozbudowane możliwości kreskowania, wymiarowania, obliczania (np. pola powierzchni), sporządzania zestawień elementów na rysunku. Dostępne są rozbudowane, ale łatwe w obsłudze funkcje edycyjne, takie jak: skracanie i wycinanie linii, rozciąganie ze zmianą wartości na liniach wymiarowych, skalowanie, obracanie, kopiowanie, zaokrąglanie, fazowanie i wiele innych. Program jest dostępny w wersji polskiej 2.5D (rysunek płaski z możliwością perspektywy) oraz w pełnej wersji przestrzennej 3D. MegaCAD efektywnie działa na komputerze od AT286 1MB RAM wzwyż i wymaga jedynie myszki.

System MegaCAD wraz z całą gamą bibliotek branżowych tworzy przyjazne środowisko pracy projektanta.

Generalny dystrybutor: CAD PO POLSKU

PIERWSZY POLSKI INTELIGENTNY
SYSTEM INFORMATYCZNY WSPOMAGAJĄCY ZARZĄDZANIE
PRODUKCJĄ - (TPPiRP), FINANSAMI - KOSZTAMI, SPRZEDAŻĄ, KADRAMI
WRAZ Z SYSTEMEM INFORMACJI KIEROWNICTWA
WSPOMAGANY SYSTEMEM SZTUCZNEJ INTELIGENCJI PC-SHELL
ORAZ EKSPERCKĄ BAZĄ WIEDZY EKONOMICZNO-FINANSOWEJ - ISAF
(DLA DUŻYCH PRZEDSIĘBIORSTW PRODUKCYJNYCH)

Perfect-Ekspert AITECH

(DZIAŁAJĄCY W OPARCIU O ŚRODOWISKO PRACY - WYŁĄCZNIE SYSTEM OPERACYJNY DOS, NetWare v3.11, v4.01 FIRMY NOVELL)

OFERUJE
PRZEDSIĘBIORSTWO PRODUKCYJNO-USŁUGOWE INNOWACJI TECHNICZNYCH

JUNISOFTEX Sp. z o.o.

ROK ZAŁOŻENIA 1987
WARTOŚĆ SPRZEDAŻY wg F-02 1992 (14,6 mld), wg F-01 na 1993.10.31 (15,4 mld)
TEL/FAX 315-131, 315-700, 321-525, 388-052, 388-054 TLX 036233
BIURO TECHNICZNO-HANDLOWE
44-100 GLIWICE ul. Kościuszki 1
WSZYSTKICH ZAINTERESOWANYCH ZAPRASZAMY NA MERYTORYCZNE
ROZMOWY DO NASZEGO BIURA TECHNICZNO-HANDLOWEGO
OFERUJEMY RÓWNIEŻ INTELIGENTNE NARZĘDZIE DO PRODUKCJI EKSPERCKICH BAZ WIEDZY

PC-SHELL

ARTIFICIAL INTELLIGENCE

DLA JEDNOSTEK BUDŻETOWYCH (NAUKA, OŚWIATA, URZĘDY MIAST I GMIN, SŁUŻBA ZDROWIA) BEZ PRAWA DO KOMERCYJNEGO WYKORZYSTYWANIA W CENIE 5.000.000,- (BEZ VAT) - INFORMACJE I ZAMÓWIENIA PROSIMY SKŁADAĆ BEZPOŚREDNIO U PRODUCENTA ARTIFICIAL INTELLIGENCE LABORATORY - AITECH, KATOWICE 14 SKR. POCZT. 2804.
DLA POZOSTAŁYCH JEDNOSTEK GOSPODARCZYCH BEZ PRAWA KOMERCYJNEGO WYKORZYSTYWANIA TJ. WYŁĄCZNIE NA WŁASNE POTRZEBY DO KOŃCA I KW. 1994 ROKU W CENIE PROMOCYJNEJ 10.000.000,- (BEZ VAT), DLA KUPUJĄCYCH DO CELÓW KOMERCYJNYCH DO KOŃCA I KW. 1994 25.000.000,- (BEZ VAT), LICENCJA OGRANICZONA NA NIEOKREŚLONĄ ILOŚĆ ODBIORCÓW KOŃCOWYCH.
POSZUKUJEMY DO WSPÓŁPRACY LUB NAWIĄŻEMY WSPÓŁPRACĘ Z EKSPERTAMI Z DOWOLNYCH DZIEDZIN NAUKI I TECHNIKI.

0/3/93

Poczta elektroniczna ICL w Brukseli

Brytyjski koncern informatyczny ICL wygrał przetarg na dostawę systemu poczty elektronicznej dla Sekretariatu Generalnego Rady Ministrów Unii Europejskiej. Konkurentami ICL były m.in. Olivetti i Hewlett-Packard. W Brukseli zainstalowany będzie TeamMail, który funkcjonuje zgodnie z protokołem X-400 i będzie współdziałać z dziewięcioma UNIX-owymi serwerami ICL DRS 6000. Komputery typu ICL DRS 6000 już od czterech lat pracują w różnych instytucjach EWG. TeamMail jest częścią systemu wspomagania prac biurowych – TeamOFFICE, który ma ponad 125 tys. użytkowników w różnych krajach. System ten funkcjonuje w ramach architektury klient-serwer a podstawowe aplikacje, takie jak edycje tekstu i arkusze kalkulacyjne są zintegrowane ze środowiskiem Windows.

Multimedialny Oracle w Anglii

Zaprezentowany w lutym br. przez Oracle Corporation zestaw wyrafinowanych narzędzi informatycznych, który pozwala na przechowywanie oraz szybkie przetwarzanie i przesyłanie informacji multimedialnych (obrazy, dźwięk wysokiej jakości, filmy, animacje itp.) będzie wykorzystywany już w tym roku nie tylko w USA, ale i w Anglii. Współpracę na tym polu z Oraclem podjął największy brytyjski koncern telekomunikacyjny – British Telecom. Na oprogramowanie to składają się: Oracle Media Server – cyfrowa „biblioteka multimedialna", która potrafi przechowywać, wyszukiwać i zarządzać różnymi formami informacji; Oracle Media Objects – narzędzie umożliwiające użytkownikom na wybieranie filmów czy zakupy z katalogu przez telefon; Oracle Media Net – zestaw służący do połączenia z serwerem multimedialnej bazy danych.

System ten umożliwi dostarczanie bezpośrednio do mieszkań całej gamy interaktywnych usług. Media Server przyda się również w biznesie, m.in. do prezentacji, seminariów, wideokonferencji itp. Podczas testów i w początkowym okresie pracy systemu, informacje cyfrowe będą przekazywane telewidzom za pośrednictwem normalnych linii telefonicznych, przy wykorzystaniu technologii Assymetric Digital Subscriber Loop (ADSL). System ma być zainstalowany w domach 60 pracowników British Telecom, a następnie u 2500 innych użytkowników. W mieszkaniach informacja będzie odbierana przez dodatkowe urządzenie przyłączone do telewizora, które zamienia go w urządzenie interakcyjne, pozwalające użytkownikom na zamawianie i odbiór informacji zgromadzonej w serwerze multimedialnym jedynie z pomocą pilota telewizyjnego.

Sun atakuje

Nowy produkt SMCC (Sun Microsystems Computer Corporation) to stacja robocza średniej mocy SPARCstation 5. Dzięki wyposażeniu w nowe procesory microSPARC II – 70MHz lub 85MHz, czterokrotnie powiększonej pamięci cache, dwukrotnie większej szybkości obliczeń zmiennoprzecinkowych i rozbudowanemu systemowi zarządzania pamięcią SPARCstation 5 jest dwukrotnie szybsza od swego poprzednika SPARCstation LX. Osiągi nowej stacji to 64 SPECint92/54.6 SPECfp92 w wersji 85MHz i 57 SPECint92/47 SPECfp92 w wersji 70MHz. Cechuje się nowoczesną konstrukcją typu „pizza box", co pozwala na niezwykłe możliwości rozszerzenia konfiguracji do 9 GB pamięci wewnętrznej, do 40 GB pamięci zewnętrznej i do 256 MB pamięci operacyjnej. System pozwala m.in. na zainstalowanie produkowanych przez SMCC kart przyspieszających przetwarzanie grafiki dwu i trójwymiarowej: TurboGX, TurboGXplus lub karty SPARCstation ZX.

Ważne są nie tylko walory użytkowe maszyny. Wygląda na to, że SMCC włączył się na dobre do toczącej się na rynku komputerowym wojny cen. Cenę nowej stacji roboczej średniej mocy Sun ustalił na zaledwie 4915 USD, zaś ceny nowej stacji roboczej dużej mocy zaczynają się już od 15 tys. USD. Poza tym firma obniżyła cenę swojego najprostszego modelu SPARCclassic do 2895 USD.

Nowe serwery IBM

IBM rozpoczął produkcję serwerów równoległych S/390 Parallel Transaction Server i S/390 Parallel Query Server, które są w stanie uruchamiać na równoległych procesorach nawet te aplikacje, które napisano dla przetwarzania szeregowego. Nowe serwery oparte są na mikroprocesorach CMOS S/390, a więc układach tego typu, jakie znajdują się w zwykłym PC. S/390 Parallel Transaction Server mogą być wykorzystane do takich aplikacji, jak identyfikacja kart kredytowych, systemy biletowe i bankowe, są także rekomendowane do sprawdzania możliwości sprzedaży i podejmowania decyzji.

Głównym elementem składowym równoległego przetwarzania transakcji jest S/390 Coupling Facility. W połączeniu z innymi cechami sprzętowymi i programowymi Coupling Facility pozwala klientom na połączenie nowych serwerów z dużymi komputerami IBM w jeden logiczny zestaw obliczeniowy – Parallel Syplex, w którym istnieje możliwość wymiany danych między komputerami. Połączone systemy mogą być tak zarządzane, jak jeden duży komputer. By umożliwić równoległe uruchamianie istniejących aplikacji, opracowano nową wersję systemu operacyjnego MVS/ESA SP, wersja 5.1. Znajduje się w niej także OpenEdition MVS – zestaw usług systemów otwartych, pozwalających na uruchamianie aplikacji UNIXo-podobnych.

Udoskonalono też funkcjonalność i wydajność systemu VM/ESA w celu rozszerzenia możliwości rozbudowy. Oprócz systemów Parallel Server IBM wypuścił także pięć podstawowych modeli procesorów ES/9000, korzystających z tych samych mikroprocesorów co systemy równoległe S/390 oraz nowy dziesięcioprocesorowy ES/9000, Model 9X2 – najpotężniejszy dostępny procesor, który ma o 22% większą moc obliczeniową niż system ES/9000 Model 982 (ośmioprocesorowy).

IBM rezygnuje z własnych klonów

W 1992 r., w ramach walki z producentami klonów (a tym samym i walki cenowej), IBM zorganizował w W. Brytanii produkcję własnych klonów – komputerów Ambra. Trafiały one również do Polski, a ich sprzedaż w całej Europie oceniono na 1% rynku. Od 1 kwietnia br. komputery te – produkowane przez filię IBM – Individual Computer Products International – nie będą sprzedawane na europejskim rynku. Jako przyczynę podano zmiany w strukturze rynku PC i stałą obniżkę cen komputerów markowych. IBM ValuePoint, PS/1 i notebook ThinkPad zaczęły przejmować – jak stwierdził W.E. McCracken, dyrektor IBM na Europę, Bliski Wschód, Azję i Afrykę – te segmenty rynku, dla których wcześniej była przeznaczona Ambra. Nie wycofano się jednak zupełnie z produkcji Ambry; komputery te są sprzedawane w USA i Kanadzie w systemie zamówień telefonicznych, który nie sprawdził się na innych rynkach. Posiadaczom Ambry, również w Polsce, serwis zapewnia IBM.

Oracle 7 wersja 7.1

Korporacja Oracle wprowadziła na rynek kolejną wersję systemu zarządzania relacyjną bazą danych – Oracle 7.1. Udostępnia ona nowe usługi: równoległe operacje i wzbogaconą replikę danych, a ponadto ułatwia programowanie na poziomie serwera oraz administrowanie bazą danych.

Opracowała:
KRYSTYNA
KARWICKA-RYCHLEWICZ